ARKEIA 5
COMMAND LINE INTERFACE MANUAL

Manual edition 5: October 2003

TABLE OF CONTENTS

Introduction

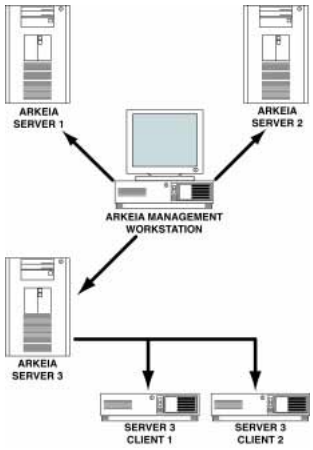
This manual presents **arkc**, the Arkeia command-line interface, as well as its different commands.

Overview

Arkeia has been designed with a three-tiered architecture, and is divided in three different parts:

1. backup server;
2. client; and
3. management/administration interface.

This allows a network administrator to fine-tune the installation of Arkeia on a network, and to administer the different Arkeia backup servers and clients from a single management workstation. The servers, in turn, will backup the different clients that have been assigned to them. This is shown in the figure below:



In the example above, the system administrator uses a single workstation, with the Arkeia interface installed, to control three different Arkeia servers. From these backup servers, he is able to backup all the clients of his network, such as the two clients connected to the Arkeia server #3 in the example. Each Arkeia server can be connected to its own backup device (tape drive, tape library, backup robot), which are not represented in the example above.

This three-tier architecture offers a great flexibility to system administrators, since they can control Arkeia servers in different locations through a single point of administration, without having to be physically present next to the backup server.

However, most of the interactions performed by the administrators is done through the Arkeia graphical user interface (GUI). While this interface has many advantages, it has the usual limitations of graphical interfaces, in the sense that it does not allow other programs to interact with Arkeia.

To allow other programs to interact with Arkeia, Knox Software has created `arkc`, which is a command-line interface to most of the functions of Arkeia. Through `arkc`, system administrators as well as other programs and utilities can access the full power of Arkeia.

Within arkc every function of Arkeia can be accessed by using a command, typed on standard UNIX command line. Therefore, the user or the utility accessing arkc can create tapes and drives, start a backup or a restoration, etc...

About this manual

This manual is designed primarily for system administrators. Knox Software recommends to every user to read through this manual, before attempting to use arkc.

This manual has been designed to be as clear as possible. To that end, it uses different typefaces to ease the presentation of information to its readers:

```
This font represents the command or the options of arkc
that should be entered by the user on a UNIX command line.
```

- This type of paragraph represents steps that should be followed to obtain a desired result or the different uses and parameters of a command.

This type of paragraph represents important information that should be read by all users. The information contained in such a paragraph could (for instance) potentially affect the proper operation of arkc or Arkeia.

This manual is divided into three parts: the first one will present the installation, the configuration the syntax, as well as the most common options of arkc. The second part will present the functions of arkc, divided by commands. The third part will present the objects (tapes, drives, etc...) that arkc can manage, along with the commands they can accept. Users starting with arkc should read the first and third part of this manual (installation, syntax and objects), while advanced users will benefit most from the second part (commands) of this manual.

4. Installation and configuration of arkc

Like the other software products released by Knox Software, arkc is available in several formats. Which format should be selected for installation depends on the architecture and operating system used.

Most UNIX architectures, such as Solaris or AIX, will use the .tar.gz (tar/gzip) package format. This package provides minimal functionality, but is based on highly standardized UNIX utilities which should be available in most systems.

Most Linux architectures, such as Red Hat or SuSE, use the .rpm (Red Hat Package Manager) package format. This package provides more functionality, such as package installation tracking, and is mainly used on Linux and Linux-based operating systems.

The rest of this chapter will provide basic installation instructions for both type of packages. For more information on the utilities or system functions involved, please refer to the manual that was supplied with your UNIX system.

Installation using .tar.gz packages

To install the `arkc` package on a system using a `.tar.gz` package, please go through the steps indicated below:

Login as the `root` user. This is usually achieved through the `su` command:

```
$ su root
```

Check the presence of the `arkc` package with `ls` command. The package name usually contains a version number, as indicated below:

```
# ls
arkc-4.1.17-1.tar.gz
...
```

Please note that the version number may be different than the one indicated above.

You should first unzip the file by entering the `gunzip` command:

```
gunzip arkc-4.1.17-1.tar.gz
```

You should now have a `tar` (archive) file named:

```
arkc-4.1.17-1.tar
```

You should untar this file with the command:

```
tar xf arkc-4.1.17-1.tar
```

To complete the installation, you should enter the directory created by the `tar` command and launch the installation script that should be present. For instance:

```
cd arkc-4.1.17-1
./install
```

The `install` script will then proceed to install the arkc software on the target machine. It will ask several questions, as well as default answers for these questions. Once the script has finished running, a valid installation should have been performed.

For instance, here is the start of the installation procedure:

```
# ./install
I1800020 main: ************* Start of installation ************
I1800020 distrbinstall: The path is /home/jack/files/bin/arkc-4.1.17-1
I1801200 mkplinstall: Installing "arkc" Version "4.1.17-1" ("Arkeia Command Line Mode")
I1801210 mkplinstall: OS: "Linux" Version "2.2.5-15"
Q1801002 *** Enter main KNOX directory [/usr/knox] ? (q to quit)
```

Installation using .rpm packages

Most Linux-based operating systems use `.rpm` (Red Hat Package Manager) packages. The main advantage of `.rpm` packages is to provide automated installation and package tracking through several command-line or graphical utilities.

To install an `.rpm` package, please follow the instructions detailed below:

Login as the root user. This is usually achieved through the su command:

```
$ su root
```

Please note that, with several X (GUI) utilities, this step may be unnecessary, as the utility itself will prompt you for the root password when launched, and perform the root login itself.

Check the presence of the arkc package with ls command. The package name usually contains a version number, as indicated below:

```
# ls
arkeia-arkc-4.1.16-1.i386.rpm
```

Please note that the version number (and architecture information) indicated could be different from the one indicated above.

Enter the command:

```
rpm --Uvh arkeia-arkc-4.1.16-1.i386.rpm
```

The RPM utility will then proceed with the installation of the software. Please note that the -U option of the RPM utility will only install a newer version of the software. For more information on the RPM utility, please refer to the man pages of your system with the command man rpm.

Standard arkc configuration

If your Arkeia configuration is simple, the easiest way to configure arkc is to edit the file named arkc.param and enter the relevant information. For instance, here is the default arkc.param file:

```
# This file is the preferences file for the Arkeia Command Line Interface.
# You have to comment out or uncomment any field which requires modification.
# Those information are used by ARKC during an arkc session.
#
# The syntax of this file must always be:
# KEYWORD"QUOTED_EXPRESSION"
#


# The NODE is your Arkeia backup server. The backup server name is a part of
# the requested information to realize a remote connection.
NODE"MyBackupServer"


# This field sets the login of the Arkeia user who executes some commands
# on the backup server. The first time you log in, you must use "root" login.
LOGIN"MyLogin"


# This field sets the password of the Arkeia user set above through LOGIN
# parameter. The first time you log in as root, you must set an empty
# password.
PASSWORD"MyPassword"


# Arkeia and Knox Software are registered trademarks, all rights reserved.
# All other company or product names mentioned are used for identification
# only and trademarks of their respective owners.
```

The content of this file should be self-explanatory. The parameters that should be entered are the name of the node (server) to be administered through `arkc`, the login used for this node, as well as the password.

Please note: Entering the user name and password in a file readable by all users may pose a security risk. Therefore, it is sound policy to protect the `arkc.param` file as much as possible. To do this, depending on the the architecture used, it may be necessary to change the ownership or readable flags of this file.

5. arkc syntax and options

The standard `arkc` syntax is the following:

`arkc -object -command [input] [output] <parameters> <filters>`

Please note that `arkc` diverge from the GNU convention of having short command names preceded by one dash: - (for instance: `-v`) and long command names preceded by two dashes: -- (for instance: `--verbose`). With `arkc`, all commands and objects should be entered fully. Please refer to the objects and commands listed below for more information.

Please note: In addition to the *canonical* syntax described above, it is also possible to use the following syntax:

arkc *-command -object* [input] [output] <*parameters*> <*filters*>

The two syntax are functionally equivalent, except the `arkc -command -object` syntax may be easier to read. Both syntax are going to be used in this manual. For instance, the two following commands are, for all intent and purposes, equivalent:

```
arkc -tape -create ...
arkc -create -tape ...
```

For more example, please refer to the rest of this manual.

arkc objects

The possible objects for `arkc` are the following ones:

- `tape`for all Arkeia operations on tapes

- `pool`for all Arkeia operations on tape pools
- `savepack`for all Arkeia operations on Savepacks
- `drive`for all Arkeia operations on drives
- `drivepack`for all Arkeia operations on Drivepacks
- `library`for all Arkeia operations on Tape libraries
- `backup`for all Arkeia commands related to backups
- `restore`for all Arkeia commands related to restore
- `journal`for all Arkeia operations on journals
- `file`for all Arkeia operations on files
- `user`for all Arkeia operations regarding users
- `debug`for debugging purposes only
- `duplication`for duplicating tapes (version *5.x* only)

arkc options: general

Help options: usage and help

At any time, it is possible to obtain an online help on an Arkeia object, by entering:

```
arkc -usage -object
```

Or by entering:

```
arkc -help -object
```

For example, if you'd like to obtain more information on the `tape` object of `arkc`, you can enter the following:

```
arkc -usage -tape
```

If you need a more detailed online help on a given command, and the way it affects an object, you can enter:

```
arkc -usage -object -command
```

To take back our `tape` example, to display additional help on the `create` command, it is possible to enter:

```
arkc -usage -tape -create
```

Help options: getinfo

If a command requires several parameters, it is possible to obtain an online help on these, by entering the following:

```
arkc -getinfo -object -command [-parameter]
```

To obtain all the parameters that will be accepted by the command `modify` and the object `tape`, we can therefore enter:

```
arkc -getinfo -tape -modify
```

Please note that the information returned by the `-getinfo` option is much more terse than the information returned by the `-usage` option. When in doubt, always use `-usage` first.

Testing and debugging options: validate

If you'd like to test the syntax of a command, without executing the arkc command itself, it is possible to enter:

```
arkc -validate -object -command [options]
```

For instance, if you'd like to test a tape creation, without really creating a tape on your Arkeia server, it is possible to enter:

```
arkc -validate -tape -create -D type=[DLT 8000] name=Test
```

The command above would test the syntax of the create command, without creating a tape.

Testing and debugging options: moreinfo

If a command fails, it is possible to obtain a detailed explanation by entering the following:

```
arkc -object -command -moreinfo
```

Testing and debugging options: verbose

If need be, all the transactions between arkc and the Arkeia server can be traced in a precise manner by entering:

```
arkc -object -command -verbose
```

Please note that -verbose option by itself does not provide additional information or online help: it simply lists the transactions that took place between arkc and the Arkeia server, as well as the results of these transactions.

For more information on the errors that can be encountered while using arkc, please refer to *Checking parameters and errors with debug* on page *56*.

Input options

With any command, arkc can accept input from three different sources: the command line itself, a configuration file, or the standard input, which allows a user to *pipe* commands into arkc.

Please note:

All the options indicated below should be entered exactly as shown. For instance, the -D option detailed below should always be entered with a capital D.

If this syntax is not respected, arkc will not process the command and display an error message.

The possible input/output options are the following:

The -D (command line) option

The syntax for this option is:

```
arkc -object -command -D [parameters]
```
The option `-D` indicates that the required parameters should appear on the command line itself. For instance, the following command will list all the available information on the tape named `foobar`:

```
arkc -tape -list -D name=foobar
```
As you can see in the example above, all the parameters that follow the option `-D` are in the format:

```
parameter=value
```
In the remainder of this manual, the different parameters will be given with a short explanation of the values they can take.

The *-I* (input file) option

The syntax of the `-I` option is the following:

```
arkc -object -command -Iinput_file
```
If the `-I` option is used, arkc will search the *input_file* for the parameters requested by the command. This file is a standard UNIX configuration file and can contain comments, if these are placed on a line that starts with the number # character.

To go back to the `-tape` example above, here is the content of an input file that creates a *file* tape:

```
# ------------------------------------------------------
# Demonstration "Input File" for arkc.
# ------------------------------------------------------

# Name of the tape
name=DemoTape

# Comment
comment=[This is a demonstration tape.]

# Type of tape
type=[FILE 50MB]

# Localization of the tape
voltag=/home/jack/arkc/files

# Recycling rules
RECYCLE_IN=SCRATCH_POOL
RECYCLE_MODE=FIFO

# ------------------------------------------------------
# This demo input file for arkc should create a "file"
# tape, with a size of 50MB, within the directory indicated
# at the "voltag=" line.
# ------------------------------------------------------
```

Please note the following points in the file displayed above:

- Whenever a value assigned to a parameter contains spaces, it must be enclosed within square brackets `[...]`. For instance, the comment and `type` parameters are enclosed within square brackets.
- Parameters are case-sensitive and predefined within Arkeia. For instance, the `type` parameter above must be entered exactly as it is displayed by the command: `arkc -tape -type`. In the example above, the `FILE 50MB` must be in capital letters and must include a space between `FILE` and `50MB`. The following values would therefore be rejected by arkc: `File 50MB`, `file 50mb`, `file50mb`.

If the rules above are not respected, `arkc` will reject the input file and

Please note that the correct syntax for the `-I` option is the following:

```
arkc -tape -create -Ifoobar.arkc
```

Whenever the option `-I` is used, there should be no space between the option and the name of the input file to be used. In the example above, the name of the file is `foobar.arkc`.

The - (standard input) option

It is possible to use the standard input to *pipe* information from other programs or commands to `arkc`. The correct syntax for this command is the following:

```
foobar | arkc -tape -create -
```

In the example above, `arkc` will create a tape, based on the information piped by the command `foobar`. Please note that the command `foobar` itself must respect the syntax used by `arkc`, including the remarks we have detailed above.

Output options

As with the input options, it is possible for `arkc` to output its messages in several different ways.

The *-O* (output file) option

The option `-O` allows the user to save the output of an `arkc` command to an external file for editing or debugging purposes. The correct syntax for this option is:

```
arkc -tape -list -D name=tape01x -Oresults.txt
```

In the example above, the information on the tape named `tape01x` is requested and the result of the command is sent to the file `results.txt`.

Please note:

When using the `-O` option, it is important to note that this option does not take into account the error messages generated by `arkc`. Error messages are always sent to the standard output.

Therefore, the command:

```
arkc -tape -list -D name=tape01x -Oresults.txt -verbose
```

would not redirect the error messages generated by `arkc` to the file named `results.txt`, assuming the tape `tape01x` does not exist on the backup server used by `arkc`.

The *-F* (filter) option

The filter option (`-F`) can be used for some functions of arkc in order to modify the information returned by the program. The correct syntax for this option is the following:

```
arkc -object -command -[D|I<input file>] <parameters...> -F<filters>
```

The filters usually take the form of a single word, that must follow the -F letter. Please note that there should be no spaces between the option itself and the filter word, and that the filter word itself should be entered exactly as shown, as it is case-sensitive. For instance, this is a valid filter option:

```
-Fbksid
```

On the other hand, the following filters options are not valid, and will produce an `arkc` error message if entered on a command line:

```
-F bksid
-F BKSID
-FBksid
```

In the following chapters, a special *filters* section will indicate which command can accept filters, and detail the usage and syntax of said filters.

Path syntax and Plugin syntax

When `arkc` requires a path to be entered anywhere in the options or commands, the path and file name entered have to respect a special syntax. This syntax is required by the Arkeia server since it is able to backup different client machines through a local area network.

The following is the standard Arkeia syntax:

```
FQDN_machine_name[!plugin_name]:/path01/.../pathX/filename
```

Here is a detailed explanation for all the parts of this syntax:

- FQDN_machine_name: This is the Fully Qualified Domain Name (FQDN) machine name. This means that arkc version 5 accepts names such as zap.foo.com, where zap is the name of the client, and foo.com the name of the domain which contains the machine named zap.

  Please note that an exclamation point ! (also called a bang! in the UNIX world) should separate the FQDN_machine_name from the plug-in name (see below).

- !plug-in_name: This is the name of the plug-in which is used by the Arkeia server to perform the backup and restoration operation. In the example screen above, the plug-in name is the default plug-in, file, which is used for all normal file system operations. Future plug-ins may be named, for instance oracle, mysql for plug-ins specialized in database backup and restoration. Please note that the plug-in name should be separated from the path by a : character.

- /path/.../filename: This is the path and the name of the file on which the operation should be performed.

Here are a few examples of correct paths/file names, that will be accepted by arkc as valid:

```
zap.foo.com!file:/home/jack/cad-files/DCCF_0124122589.dat
zip.foo.com!file:/home/tina/docs/pdf/arkeia/man_arkc.pdf
email01.neon.com!file:/etc/mail/statistics/
```

Please note: The syntax descrived above applies only to arkc version 5 or above!! This syntax is valid whether the machine is a Windows or a UNIX machine. Arkeia uses slash characters / for paths on all operating systems.

The only difference in syntax between the Windows and the UNIX Arkeia client is that Windows machines require the volume letter to be inserted before the path and file name.

Therefore the following paths are correct only for Windows machines:

```
bilbo.shire.org!file:c:/files/docs/linux/pdf/LinuxFromScratch.pdf
```

6. The create command of arkc

The arkc create command is used to create the following objects on an Arkeia server:

- tape
- pool

- drive
- savepack
- drivepack
- user
- library
- tape duplication

We are going to detail each object that can be created with this command in the rest of this chapter. Each object will include the different options that can be applied to its creation.

Please note: The creation of certain objects, especially the creation of libraries, requires a correct license of Arkeia to be installed on the Arkeia server.

If the correct license is not installed, arkc will refuse the creation of the objects that are not covered by the current license.

Tape creation

The creation of a tape with arkc, through the create command, uses the following syntax:

```
arkc -tape -create -[D|I] <parameters>
```

Please refer to the section named *arkc options: general* on page *10*, for more information on the different options that can be used with arkc.

The parameters that can be used with the create command are the followings:

- name(**Required**)
  The name parameter indicates the name which is going to be used by Arkeia to refer to the new tape. This parameter is required by the create command. The correct syntax for this parameter is: name=XXXXX, where XXXXX is the name (string) to assign to the new tape.
  For instance:

  ```
  name=DemoTape
  name=DAT_Database
  ```

- type(**Required**)
  The type parameter indicates the type of the tape being created. This parameter is required by the create command. Please note that the names should be entered precisely as shown below and that, whenever a name contain spaces, it should be enclosed with square brackets [...], after the type= function. For instance: type=[EXB 8500] is a valid syntax, while type=exb 8500 or type=Exb8500 are not. The available tape types are listed below:

```
AIT-1
AIT-1XL
AIT-2
DAT-DDS4
DAT-DDS3
DAT-120
DAT-60
DAT-90
D3-10GB
D3-25GB
D3-50GB
EXB 8500
EXB 8505XL
EXB MAMMOTH
EXB MAMMOTH 2
NULL
FILE 20MB
FILE 50MB
FILE 100MB
FILE 500MB
DLT 2000
DLT 2000XT
DLT 4000
DLT 7000
DLT 8000
DLT1
SDLT
MAGSTAR MP
MAGSTAR MP-C
MAGSTAR MP-CXL
DTF GW 730L
QIC 50GB
QIC 25GB
QIC 16GB
QIC 13GB
QIC 4GB
QIC 2GB
3590 CART
9840 CART
LTO Ultrium
VXA-V17
VXA-V6
```

Please note: It is possible to obtain this list of available tape types by entering the following command:

```
arkc -tape -type
```

For more information about the `type` command, please refer to the chapter named *The informational commands and objects of arkc* on page *48* of this manual.

- `comment`(Optional)
  The `comment` parameter can be used to insert a short string to offer more information to the user about the tape. The comment is displayed within the Arkeia GUI, or with the `-list` command of `arkc` (Please see the command `-list`, below, for more information). The correct syntax for this parameter is as follows:

```
comment=[This tape is for database backup only]
```

Please note that a comment containing spaces should be placed between square brackets `[...]` as shown above.

- `firstnum` ... `lastnum`(Optional)
  `firstnum` and `lastnum` allow the user to define several identical tapes with only two commands. To do this, it's only necessary to enter the number of the first tape (after `firstnum=`) and the number of the last tape (after `lastnum=`). For instance, the following entries in a data file would create five (5) tapes:

  ```
  # Create 5 tapes _________
  firstnum=1
  lastnum=5
  ```

  Please note that the name of the tapes will be the value assigned to the command `name=`, followed by the tape number. The value of `firstnum` can be different than 1.

- `plname`/`plid`(Optional)
  The parameters `plname` and `plid` indicates to which tape pool the new tapes should be added. The parameter `plname` indicates the pool chosen by its name, and `plid` indicates the pool chosen by its internal Arkeia ID. For instance:

  ```
  plname=Test_Pool
  ```

  In the example above, the tape created with `-tape -create` will be added to the tape pool named `Test_Pool`.

- `recycle_in`(Optional)
  The parameter `recycle_in` determines in which tape pool the tape created should be placed when it is recycled. There are two possible pools defined by default: `SCRATCH_POOL` (default recycling pool) and `CURRENT_POOL`. For instance, the parameter:

  ```
  recycle_in=CURRENT_POOL
  ```

  will recycle the tape created in example above to `Test_Pool`, if the tape was assigned to this pool when it was created. By default, the tape is recycled in the pool it was assigned to when it was created.

- `recycle_mode`(Optional)
  The parameter `recycle_mode` determines the type of recycling operation that will be applied to the tape being created. This parameter can take two value: `FIFO` (first in/ first out) or `LIFO` (last in/first out). If the value chosen is `FIFO`,

the oldest recycled tape will be used for a backup. If the value is LIFO, the newest recycled tape is used for the backup. For instance:

```
recycle_mode=FIFO
```

Please note that the values of this parameter should be entered all in uppercase letters. The default value is FIFO.

- access(Optional)
  The parameter access defines the access rights for the tape created. The possible access rights are the following: WRITE, READ, RECYCLE, DELETE, CLEAN. The different rights are defined within the same access parameters as follows:

  ```
  access=[WRITE|READ|RECYCLE]
  ```

  The parameter above authorize the WRITE, READ and RECYCLE operations on the tape that was created, but not any other. The default rights on all tapes are: READ, WRITE, RECYCLE, and DELETE.

- voltag(Optional)
  The parameter voltag is applied only to the tape of the type FILE (see above for tape types. The value of this parameter is the path which will be used to create the file itself. For instance:

  ```
  # Directory for the tape
  voltag=/home/jack/arkeia/files/arkeia_file
  ```

  The parameter above will create an Arkeia FILE tape in the directory: /home/jack/arkeia/files/arkeia_file. Please note that this parameter is required by all FILE tapes. You cannot put more than one tape in a given directory.

Pool creation

The creation of a tape pool with arkc, through the create command, uses the following syntax:

```
arkc -pool -create -[D|I] <parameters>
```

Please refer to *arkc options: general* on page *10*, for more information on the different options that can be used with arkc. The parameters that can be accepted by the create command when it is used to create a new tape pool are the following:

- name(**Required**)
  The name parameter indicates the name which is going to be used by Arkeia to refer to the new tape pool. This parameter is required by the create command. The correct syntax for this parameter is: name=XXXXX, where XXXXX is the name (string) to assign to the new pool. For instance:

  ```
  name=Safe_Pool
  ```

creates a tape pool named: Safe_Pool. Please note that it is possible to assign a name that contains space, if this name is entered between square brackets: [...]. For instance:

```
name=[R&D tape pool]
```

The parameter above will create a Tape Pool named: R&D tape pool.

- owner(Optional)
  The owner parameter indicates which Arkeia user is the owner of the tape pool. By default, the value of owner is set to the user who opened the session. It is possible to specify a different user than the user currently logged in to the session. For instance, if the command:

  ```
  $ arkc -user -list
  ```

  returns the following list of users:

  ```
  name=jsmith(John Smith)
  name=tchang(Tina Chang)
  name=root
  ```

  The owner value can be either jsmith, tchang or root.

- comment(Optional)
  The comment parameter can be used to insert a short string to offer more information to the user about the tape pool being created. The comment is displayed in the Arkeia GUI, or with the -list command of arkc (For more information on this command, please refer to *Listing objects with the list command* on page *49*). The correct syntax for this parameter is as follows:

  ```
  comment=[This group of tape should be placed in the safe]
  ```

  Please note that comment containing spaces should be placed between square brackets [...] as shown above.

Drive creation

The creation of a tape drive with arkc, through the create command, uses the following syntax:

```
arkc -drive -create -[D|I] <parameters>
```

The parameters that can be used while creating a tape drive are the following ones:

- name(**Required**)
  The name parameter indicates the name which is going to be used by Arkeia

to refer to the new tape drive. This parameter is required by the `create` command. The correct syntax for this parameter is: `name=XXXXX`, where `XXXXX` is the name (string) to assign to the new drive. For instance:

```
name=Accounting
```

The parameter above creates a drive named: `Accounting`.

- `type`(**Required**)
  The `type` parameter indicates which type of tape drive is going to be created by arkc. Please note that this parameter is required by the `create` command. Here is a list of available types of tape drive that are valid for this parameter:

  ```
  NULL
  FILE
  STD_AIT
  STD_DAT
  STD_EXABYTE
  STD_DLT
  STD_DTF
  DAT_SONY_7000
  AIT_AUTOLOADER
  STD_QIC
  STD_CARTRIDGE
  REDWOOD_SD3
  LTO
  MAGSTAR_MP
  ECRIX_VXA
  9840
  ```

  For instance, to create a DLT tape drive, a correct command syntax would be the following one:

  ```
  arkc -drive -create -D name=DLT02 type=STD_DLT
  rewind_dev=/dev/st0
  ```

  Please note that this command should be entered all on one line. You can also refer to the parameter `rewind_dev` below).

- `rewind_dev`(**Required**)
  The parameter `rewind_dev` is used to define the system device used by the operating system to access the tape drive. This device is usually used by the `mt` UNIX command. For instance, under most Linux systems, this device is `/dev/st0`.

- `access`(Optional)
  This parameter is used to define the access rights to be assigned to the device being created. These rights can be `READ`, `WRITE`, `RECYCLE` and `DELETE`. To authorize certain access rights, these should be included as values of the `access`

parameter, for instance:
`access=[READ|WRITE]`
The parameter above indicates that the drive being created can be accessed for read/write operations, but not for deleting or recycling. By default, all four type of rights are granted on a new tape drive.

- `comment`(Optional)
  This parameter is used to assign a descriptive comment to the tape drive being created. This comment can be useful to display additional information related to the tape drive, as, for instance, its location, purpose, etc. The correct syntax for this parameter is:
  `comment=[DLT tape drive for CAD machines]`

Savepack creation

Within Arkeia, a *Savepack* is a group of files, directories, disks or even whole set of machines, which is used to simplify a backup operation. Instead of choosing individual files, an administrator can create a Savepack, name it, and use the Savepack to quickly and effortlessly create and launch repetitive backup operations.

To create a Savepack with `arkc` and its `create` command, use the following syntax:

```
arkc -savepack -create -[D|I|-] <parameters>
```

Please note: using `arkc -create -savepack` only creates an empty Savepack. To add files and trees to a new Savepack, it is necessary to use the command `arkc -savepack -modify` (See below).

The parameters that can be used while creating a Savepack are the following ones:

- `name`(**Required**)
  The `name` parameter assign a name to the Savepack that will be created by arkc. This parameter is required by the `create` command. The correct syntax for this parameter is: `name=XXXXX`, where `XXXXX` is the Savepack's name. For instance:

  ```
  name=Accounting
  ```

  The parameter above creates a Savepack named: `Accounting`. If you'd like to insert spaces in the name of the Savepack, you should enter the name between square brackets `[...]`. For instance:

  ```
  name=[CAD/CAM Savepack]
  ```

- `comment`(Optional)
  This parameter is used to assign a descriptive comment to the Savepack being created. This comment can be useful to display additional information. The correct syntax for this parameter is:

```
comment=[Savepack for CAD machines]
```

- `allowed_fs`(Optional)
  The parameter `allowed_fs` defines how Arkeia is going to process different file systems while backing up the Savepack. This parameter can accept the three following values:

  ```
  NORMAL
  ALL
  ALL_EXCEPT_NFS
  ```

  When `allowed_fs` is set to `NORMAL`, the back-up of a Savepack will only process normal file systems. The value `ALL` indicates that all file systems (local, NFS and others) should be backed-up. The value `ALL_EXCEPT_NFS` indicates that all file systems, except NFS, should be backed-up while Arkeia is backing-up a given Savepack.

- `retry`(Optional)
  This parameter defines the number of times Arkeia should try to connect to a remote machine, that contains a tree included in the Savepack being backed-up. By default, the value of `retry` is `3`. The possible values are numbers from `1` through `10`.

- `compress`(Optional)
  This parameter defines what type of compression should be applied to the files contained in the Savepack. The possible values for `compress` are the following:

  ```
  LZ1_LZ3
  LZ1
  LZ3
  NO_COMPRESS
  ```

  The default value for the `compress` parameter is `LZ1_LZ3`, which lets Arkeia choose the optimal compression scheme between `LZ1` and `LZ3`, based on the data contained in the file being backed-up. The value `NO_COMPRESS` turns the file compression function off.

- `crypt`(Optional)
  This parameter defines the type of encryption that will be applied to the files contained in a Savepack. The possible values for `crypt` are the following:

  ```
  DES_BLOWFISH
  DES
  BLOWFISH
  NO_CRYPT
  ```

By default, the value of `crypt` is `DES_BLOWFISH`. Using the value `NO_CRYPT` turns off the encryption features of Arkeia. The encryption functions that are available within Arkeia are described in much more details in the Arkeia User Manual, which is available on Knox Software web site (http://www.arkeia.com).

- `symb_link`(Optional)
  The parameter `symb_link` determines the behavior of Arkeia when it encounters a UNIX symbolic link within the files contained in a Savepack. The possible values for this parameter are `YES` or `NO`. A value of `YES` indicates that Arkeia should follow the symbolic links and a value of `NO` indicates that symbolic links should not be followed. The default value for this parameter is `NO`.

Please note: The `symb_link` parameter should be used with caution, as symbolic links can sometimes reference each other in a loop. For instance, link `A` can point to link `B`, which, in turn, points back to link `A`. Such a situation creates an infinite loop, which will block Arkeia's normal operations. Caution is therefore strongly advised.

- `follow_fs`(Optional)
  The parameter `follow_fs` determines the behavior of Arkeia when it encounters a mounted file system contained within a Savepack to be backed-up. This parameter can accept two values: `YES` (back-up mounted file systems) and `NO` (do not back-up mounted file systems). By default, the value of `follow_fs` is `YES`.

- `inc_filter`(Optional)
  The `inc_filter` defines an inclusion filter that will be applied to the backup of a Savepack. Once a filter is defined, Arkeia will backup the Savepack files only if they match the filter. All other files will be ignored. The filter itself is a normal UNIX regular expressions. For instance:

  ```
  inc_filter=.*W..\.dat
  ```

  The filter above will only backup the files whose name contain an upper-case `W`, followed by any two characters and by the letters `.dat`. For instance, a file named `CAD_Project.W55.dat` would be backed-up with such a filter.

- `exc_filter`(Optional)
  The `exc_filter` defines an exclusion filter that will be applied to the backup of a Savepack. Once a filter is defined, Arkeia will not backup the Savepack files if they match the filter. All other files will be saved. The filter itself is a

normal UNIX regular expressions. For instance:

```
exc_filter=.*D..\.dat
```

The filter above won't backup the files whose name contain an upper-case `W`, followed by any two characters and by the letters `.dat`. For instance, a file named `DataBase.D56.dat` would not be backed-up with such a filter in place.

- `cmd_before`(Optional)
  The parameter `cmd_before` defines a command that has to be executed before the Savepack is processed by Arkeia. Please note that the command will be executed on the client machine. This parameter accepts any command that can be executed, including the path if necessary. For instance:

  ```
  cmd_before=[rm -fv /home/jack/temp/*.tmp]
  ```

  This command will remove all files ending with `.tmp` in the directory `/home/jack/temp/` prior to the backup of the directory `/home/jack` contained in the Savepack.

- `do_cmd_before`(Optional)
  The parameter `do_cmd_before` defines the behavior of Arkeia if the command defined by the parameter `cmd_before` fails. This parameter can accept two values: `YES` or `NO`. If `do_cmd_before` is set to `YES`, the back-up will proceed even if the command failed. If it is set to `NO`, the backup of the Savepack will be aborted if the command launched before the backup fails.

- `cmd_after`(Optional)
  The parameter `cmd_after` defines a command that has to be executed after the Savepack is processed by Arkeia. Please note that the command will be executed on the client machine. This parameter accepts any command that can be executed, including the path if necessary. For instance:

  ```
  cmd_after=[dvrt -xc]
  ```

  This means that the command `dvrt` will be executed once all the files contained in the Savepack have been backed-up.

- `do_cmd_after`(Optional)
  The parameter `do_cmd_after` defines the behavior of Arkeia if the file backup fails. This parameter can accept two values: `YES` or `NO`. If `do_cmd_after` is set to `YES`, the command defined by the parameter `cmd_after` will always be executed, even if the backup itself fails. If it is set to `NO`, the command defined by `cmd_after` will not be executed if the backup of the Savepack has failed.

Drivepack creation

A Drivepack is a group of drives that is created within Arkeia in order to ease the management of large backups that may span several tapes/drives. Drivepacks allow the user to group drives and use them, as a single entity, for all backup operations.

Drivepacks are created with the following command:

```
arkc -create -drivepack -[D|I|-] <parameters>
```

The parameters that can be accepted while creating a Drivepack are the following:

- `name`(**Required**)
  The `name` parameter assign a name to the Drivepack that will be created by arkc. This parameter is required by the `create` command. The correct syntax for this parameter is: `name=XXXXX`, where `XXXXX` is the Drivepack's name. For instance:

  ```
  name=3rd_Floor
  ```

  creates a Drivepack named: `3rd_Floor`. If you'd like to insert spaces in the name of the Drivepack, you should enter the name between square brackets `[...]`. For instance:
  `name=[DLT Drivepack]`

- `comment`(Optional)
  This parameter is used to assign a descriptive comment to the Drivepack being created. This comment can be useful to display additional information. The correct syntax for this parameter is:

  ```
  comment=[DAT Drivepack for small backups]
  ```

- `nbdrive`(Optional)
  This parameter defines the number of drives that can be used by the Drivepack for a given operation. By default, the value of `nbdrive` is the number of drives that are contained in the Drivepack. If `nbdrive` has a lower value, this means a group of drives within the Drivepack will be used for the first task, while some drives will be reserved for other tasks. For instance, if a Drive pack contains 5 drives, it is possible to reserve three of these (`nbdrive=3`) for backup and keep 2 drives for restoration operations. Combining this parameter with the `priority` parameter (see below for more information) offers a great flexibility to system administrators in deciding which drive will be used for what purpose.

  Please refer to the information below (`drvname` and `drvid` parameters) for more details on adding new drives to a Drivepack.

- drvname(Optional)
  To add drives to a Drivepack, it is possible to use either the drvname or the drvid parameters. Both these parameters should be contained in an input file, with each drive to be added to the Drivepack contained in a pair of <ITEM>...</ITEM> statements. For instance, the following lines of an input file would add the drive named DLT01 in a Drivepack:

  ```
  # Comment: First DLT drive of drivepack.
  <ITEM>
  drvname=DLT01
  priority=1
  </ITEM>
  ```

  Adding several drives to a Drivepack is therefore a question of adding several <ITEM>...</ITEM> statements into the same input file, each statement containing different drvname and priority parameters.

- drvid(Optional)
  To add drives to a Drivepack, it is possible to use either the drvname or the drvid parameters. Both these parameters should be contained in an input file, with each drive to be added to the Drivepack contained in a pair of <ITEM>...</ITEM> statements. For instance, the following lines of an input file would add the drive with the ID of DLT01 in a Drivepack:

  ```
  # Comment: First DLT drive of drivepack.
  <ITEM>
  drvid=3bc419c3
  priority=1
  </ITEM>
  ```

  Adding several drives to a Drivepack is therefore a question of adding several <ITEM>...</ITEM> statements into the same input file, each statement containing different drvname and priority parameters.

Please note: It is possible to add drives to a Drivepack by using the -D option, followed by the drvname or the drvid parameters. But using the command line does not allow the use of a specific priority parameter for each drive. In most cases, it is highly recommended to use an input file containing the drive parameters, with the -I command-line option, in order to set priority for each specific drive.

For example, the following line will create a new Drivepack based on the Create_Drivepack.dat input file:

- priority(Optional)
  The parameter priority assigns a priority to a drive contained in a Drivepack. This allows an administrator to define precisely which drive will receive first

the data from a back-up operation. For instance, if drive A has a priority of 1 and drive B has a priority of 5, the A drive will be used first for a backup operation. If this operation requires more drives, then drive B will be used, followed by any other drives in the order of their priority.

This parameter can accept values from 1 to 128, with a value of 1 (highest priority) being assigned by default.

Combining the priority parameter with the nbdrive offers a high level of flexibility for the operation of a group of drives. Please refer to the above information for more examples on how to add this parameter to a Drivepack input file.

User creation

Creating a user can be accomplished with the create command with the following syntax:

```
arkc -user -create -[D|I] <parameters>
```

The parameters that can be used to create a user are the following ones:

- name(**Required**)
  The parameter name assigns a descriptive name to the user being created. The correct syntax for this parameter is, for example:

  ```
  name=John
  ```

  If a name must contain spaces, it should be enclosed between square brackets: [...]. For instance:

  ```
  name=[John Smith]
  ```

- node(**Required**)
  The parameter node contains the name of the Arkeia client the user will interact with. This parameter has the following syntax:

  ```
  node=stormbringer
  ```

  In the example above, the name of the Arkeia client which is to be used is stormbringer. A node should always be assigned to a newly-created user.

- email(Optional)
  The email parameter is used to define an email address that Arkeia can use to warn the user of the completion of an operation. The email address should be a standard address, that can be reached by the Arkeia server. For instance:
  email=john.smith@foobar.com

- passwd(Optional)
  The passwd parameter defines the password used by the user to connect to the Arkeia server defined by the node parameter. It is highly recommended to use secure UNIX passwords to avoid security problems. For instance:

  ```
  passwd=DF56AX82VV08
  vpasswd=DF56AX82VV08
  ```

Please note: the password defined by the passwd parameter should always be confirmed with the vpasswd parameter, as in the example above.

Library creation

Arkeia is able to manage complex backup hardware, such as large tape libraries. This type of hardware should be defined within Arkeia in order to be used for back-up and restore operations.

The standard syntax to create a library with arkc is the following:

```
arkc -library -create -[D|I] <parameters...>
```

The parameters that can be accepted by arkc while creating a library are the following ones:

- name(**Required**)
  The parameter name assigns a descriptive name to the library being created. This parameter is mandatory. For instance:

  ```
  name=DLT_Lib04
  ```

  or:

  ```
  name=[Accounting DLT library]
  ```

- type(**Required**)
  The parameter type defines the type of library being created. Please note that the value of type must be entered precisely as shown. Some possible values for this parameter are:

  ```
  ADIC100-LTO
  ATL-L200
  AUTOPAK
  BLACKJACK_X_21
  DLT4700
  DLT7
  DLTSTOR114
  EXB-17D7
  EXB-60
  EXB-EZ17
  ```

```
EXB-X200
FALCON-1530
FALCON-1560
FASTSTOR
FILE
HP19-LTO
HP660-LTO
HP660
HPC1191
HPDAT6
Harrier8150
Harrier830
Harrier850
IBM7336
IBM7337
ITL2225
ITL4115
ITL7115
LIBRA16
LIBRA18
LIBRA8
LP1117
MAGSTAR_L12
MAGSTAR_L18
MAGSTAR_L24
MAGSTAR_L32
SONY_DMSB9
STD4586NP-12
STD4586NP
STK7430
STK9710
STK9714
SUN-L1000
SUN-L700-9840
SUN-L700-DLT
SUN-L9
TSL-A300
TSL-S7000
TWINLIBRA16
TWINLIBRA18
```

You can obtain a complete list of all the hardware supported by Arkeia by typing the command: `arkc -library -type` on a command line. For more information, please see chapter 6: The informational commands of arkc.

Please note: The value of `type` should be entered exactly as shown in the list above. These values are case-sensitive. Therefore, the value `hp19-dlt` and `HP19-DLT` are not equivalent.

A library creation requires the appropriate license of Arkeia. If you do not have a license for a library, `arkc` will produce an error message and exit. For more information on the Arkeia licensing scheme, please contact: `sales@arkeia.com`.

- `libdev`(**Required**)
  The parameter `libdev` indicates which UNIX device is the controller of the library being created. This device will vary according to the system and its configuration. For instance, here are two possible values for `libdev`:

  `libdev=/dev/sg0`(Linux)
  `libdev=/dev/rsst4`(Sun Solaris)

  For more information on control devices, please refer to the manual of your UNIX/Linux system, and to the output of the `dmesg` UNIX command.

- `comment`(Optional)
  The parameter `comment` assigns a descriptive comment to the library being created. This parameter is useful to display more information on a given library. For instance, the following is a valid comment for `arkc`:

  ```
  comment=[DLT library for Accounting Department]
  ```

  Please note that, as you can see above, comments that contain spaces should be included between square brackets: `[...]`.

Tape duplication creation

One of the newest function of `arkc` is the tape duplication. By creating a tape duplication object, it is possible to make a perfect copy of the most important Arkeia tapes and of the data they contain. Once created, a `duplication` object can be started to make a copy of a given tape, to keep the copy in a separate location in case the original tape is lost or damaged.

Important information:

Please note that the tape duplication functions are only available in version 5.x (or older) of `arkc`.

Previous versions of `arkc` do not offer the tape duplication func-

The normal syntax of the tape duplication is the following:

```
arkc -duplication -start -D <parameters...>
```

The `-D` parameter indicates to `arkc` that its arguments and parameters are entered on the command line.

The following parameters must be entered on the command line, when duplicating a tape from one drive to another:

- `src_tape_name` or `src_tapeid`: (**Required**)

  These parameters are required by `arkc`.

These parameters should contain either the name or the ID of the source tape. For instance, the following names and IDs are considered valid, and will be accepted by `arkc` for a tape duplication command:

```
src_tape_name=DLT004
src_tapeid=3bc1d2e2
```

As can be seen above, the `src_tape_name` may be easier to remember, but both these values are valid, and accepted, by `arkc`. To display a list of available source tapes, enter the command: `arkc -tape -list` (without the quotes). The tape name is assigned when a tape is created by Arkeia.

- `src_drv_name` or `src_drvid`: (**Required**)

  These parameters are required by `arkc`.

  These parameters should contain either the name or the ID of the source drive. For instance, the following names and IDs are considered valid, and will be accepted by `arkc` for a tape duplication command:

  ```
  src_drv_name=RD_LTO
  src_drvid=3a530bcf
  ```

  As can be seen above, the `src_drv_name` may be easier to remember, but both these values are valid, and accepted, by `arkc`. To display a list of available drives, enter the command: `arkc -drive -list` (without the quotes).

- `dst_drv_name` or `dst_drvid`: (**Required**)

  These parameters are required by `arkc`.

  These parameters should contain either the name or the ID of the destination drive. This drive must be compatible with the source drive! For instance, the following names and IDs are considered valid, and will be accepted by `arkc` for a tape duplication command:

  ```
  dst_drv_name=RD_LTO
  dst_drvid=3a530bcf
  ```

  As can be seen above, the `dst_drv_name` may be easier to remember, but both these values are valid, and accepted, by `arkc`. To display a list of available drives, enter the command: `arkc -drive -list` (without the quotes).

Additionally, the following parameters can be entered, when duplicating a tape that is contained in a tape library:

- `dst_lib_name` or `dst_libid`: (**Required**)

  These parameters are required when using `arkc`, if duplicating in a library.

These parameters should contain either the name or the ID of the destination library. This library must contain the destination drive defined above! For instance, the following names and IDs are considered valid, and will be accepted by `arkc` for a tape duplication command:

```
dst_lib_name=RD_LTO
dst_libid=3a530bcf
```

As can be seen above, the `dst_lib_name` may be easier to remember, but both these values are valid, and accepted, by `arkc`. To display a list of available libraries, enter the command: `arkc -library -list` (without the quotes). Please note that defining `dst_lib_name` or `dst_libid` means that `dst_slot` (see below) must be defined as well.

- `dst_slot`: (**Required**)

  This parameter is required when using `arkc`, but only if duplicating in a library.

  This parameter should contain the number of the slot that contains the destination tape. This parameter must be defined if duplicating to a library! For instance, the following is considered as valid value, and will be accepted by `arkc` for a tape duplication command:

  ```
  dst_slot=6
  ```

  This slot must contain the tape that will be used as a target by the duplication operation! Please note that, when duplicating from/to a library, it is not necessary to define a source slot for the source tape, as Arkeia is able to retrieve the correct tape based on its ID and/or name. Also, please note that defining `dst_slot` means that `dst_lib_name` or `dst_libid` (see above) must be defined as well.

7. The delete command of arkc

The `delete` command of `arkc`, while important, is usually much more simple than the `create` command. Its function is to delete objects from the Arkeia configuration.

Its general syntax is:

```
arkc -[object] -delete -[D|I] <parameters>
```

Where `parameters` is usually `name`, followed by the full name that has been assigned to the object when it has been created.

The rest of this chapter will go into more details on the objects that can be deleted.

Tape deletion

To delete a date with `arkc`, use the following syntax:

```
arkc -tape -delete -D <parameters>
```

Please note: Deleting an Arkeia tape does not affect the physical tape itself, it means that the entry of the tape in the Arkeia database will be deleted. The tape data won't be accessible anymore, since the Arkeia database will not have a record of the tape.

The only way to access the tape data again is to import the tape information back in the Arkeia database with the `arkrstb` utility, an operation which can be complex and time-consuming if several tapes have been deleted. For more information on this operation, and the utilities provided to perform it, please refer to the Arkeia

The following parameters can be used to delete a tape:

- `name`(**Required**)
  The `name` parameter indicates which tape should be deleted by `arkc`. For instance, the following command will delete the tape named `Test_Tape_01`:

  ```
  arkc -tape -delete -D name=Test_Tape_01
  ```

  To delete several tapes, you can either:

  (**a**) Enter several names on the command line:

  ```
  arkc -tape -delete -D name=Test_Tape_01 name=Test_Tape_02
  ```

  (**b**) Delete all the tapes contained in a pool of tape (if all the tapes to be deleted are assigned to the same pool):

  ```
  arkc -tape -delete -D plid=3a78354e
  ```

  For more information on tape pool deletion, please see the parameter `plid` below.

  or (**c**) perform a piping operation, such as:

  ```
  arkc -tape -list | grep Test_ | arkc -tape -delete -
  ```

  In the command above, `arkc` first makes a list of tapes, which is piped to, and parsed by, the UNIX `grep` command. In the example above, all tapes with a name starting with `Test_` will be returned. The result is then piped back into `arkc` for deletion. Please note the usage of the `-` input option in this example.

- `tpid`(Optional)
  The parameter `tpid` can replace the `name` parameter that we have detailed above. The `tpid` value is assigned to a tape when it is first created by Arkeia. This value can be displayed for a given tape by entering the command:

  ```
  arkc -tape -list -D name=name_of_tape
  ```

  Since the `tpid` parameter is equivalent to the `name` parameter, the two commands below are equivalent (provided that `3a7834d7` is the correct `tpid` value of the tape `Test_Tape_01`):

  ```
  arkc -tape -delete -D name=Test_tape_01
  arkc -tape -delete -D tpid=3a7834d7
  ```

- `plid`(Optional)
  The `plid` parameter is one way to delete several tapes in one operation, as long as all the tapes to be deleted are included in the same tape pool. The `plid` of a given tape can be obtained by entering the following command:

  ```
  arkc -tape -list -D name=name_of_the_tape
  ```

  Once obtained, it is possible to delete all the tapes contained in a given pool tape by entering the command:

  ```
  arkc -tape -delete -D plid=3a78354e
  ```

  In the example above, all the tapes contained within the tape pool ID `3a78354e` will be deleted with one command.

Please note: Caution is advised when using the `plid` parameter while deleting several tapes, as the `delete` command will delete all the tapes contained in the tape pool. For more information on tape pool deletion, please see below.

Tape pool deletion

Deleting a tape pool is almost the same as deleting a single tape.

The correct syntax to delete a tape pool with `arkc` is the following one:

```
arkc -pool -tape -D [name|plid]=<value>
```

The parameters that can be accepted while deleting a tape pool are the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the pool to be deleted. For instance:

```
arkc -pool -delete -D name=Test_Pool
```

A list of all the tape pools available on a given Arkeia server can be obtained with the command:

```
arkc -pool -list
```

- `plid`(Optional)
  The `plid` parameter is the ID assigned to the tape pool when it was created by Arkeia. This ID value can be used in place of the `name` parameter described above. For instance, the two commands below are equivalent, assuming that `3a78354e` is the correct ID of `Test_Pool`:

```
arkc -pool -delete -D name=Test_Pool
arkc -pool -delete -D plid=3a78354e
```

Please note: To delete a tape pool, it is necessary to first delete all the tapes that it contains. A tape pool which is not *empty* of tapes cannot be deleted by `arkc`. Any attempt to delete a non-empty tape pool will produce an error message.

Drive deletion

Deleting a drive allows the system administrator to quickly reconfigure Arkeia, in the case maintenance or replacement is needed for a hardware unit.

The correct syntax to delete a drive is the following one:

```
arkc -drive -delete -[D|I] <parameters...>
```

Where the parameters can be the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the drive to be deleted. For instance:

```
arkc -drive -delete -D name=DLT02
```

A list of all the tape drives available on a given Arkeia server can be obtained with the command:

```
arkc -drive -list
```

- `drvid`(Optional)
  The `drvid` parameter is the ID assigned to the tape drive when it was created

by Arkeia. This ID value can be used instead of the `name` parameter described above. For instance, the two commands below are equivalent, assuming that `3bc419cd` is the correct ID of the tape drive named `DLT02`:

```
arkc -pool -delete -D name=DLT02
arkc -pool -delete -D plid=3bc419cd
```

Savepack deletion

The correct syntax to delete a Savepack is the following one:

```
arkc -savepack -delete -[D|I] <parameters...>
```

Where the parameters can be the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the Savepack to be deleted. For instance:

  ```
  arkc -savepack -delete -D name=WebDesign
  ```

  A list of all the savepacks available on a given Arkeia server can be obtained with the command:

  ```
  arkc -savepack -list
  ```

- `skid`(Optional)
  The `skid` parameter is the ID assigned to the Savepack when it was created by Arkeia. This ID value can be used instead of the `name` parameter described above. For instance, the two commands below are equivalent, assuming that `3a64332e` is the correct ID of the Savepack named `WebDesign`:

  ```
  arkc -savepack -delete -D name=WebDesign
  arkc -savepack -delete -D plid=3a64332e
  ```

Drivepack deletion

The correct syntax to delete a Drivepack is the following one:

```
arkc -drivepack -delete -[D|I] <parameters...>
```

Where the parameters can be the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the Drivepack to be deleted. For instance:

  ```
  arkc -drivepack -delete -D name=DLT_drivepack
  ```

  A list of all the Drivepacks available on a given Arkeia server can be

obtained with the command:

```
arkc -drivepack -list
```

- dkid(Optional)
  The dkid parameter is the ID assigned to the Drivepack when it was created by Arkeia. This ID value can be used instead of the name parameter described above. For instance, the two commands below are equivalent, assuming that 3bc46702 is the correct ID of the Drivepack named DLT_drivepack:

  ```
  arkc -drivepack -delete -D name=DLT_drivepack
  arkc -drivepack -delete -D plid=3bc46702
  ```

User deletion

The correct syntax to delete an Arkeia user is:

```
arkc -user -delete -[D|I] [name=User_Name]
```

Where the parameters can be the following ones:

- name(**Required**)
  The name parameter should contain the name of the user to be deleted. For instance:

  ```
  arkc -user -delete -D name=John
  ```

  A list of all the users available on a given Arkeia server can be obtained with the command:

  ```
  arkc -user -list
  ```

Library deletion

The correct syntax to delete an Arkeia library is the following one:

```
arkc -library -delete -[D|I] <parameters...>
```

Where the parameters can be the following ones:

- name(**Required**)
  The name parameter should contain the name of the library to be deleted. For instance:

  ```
  arkc -library -delete -D name=Lib04
  ```

  A list of all the libraries available on a given Arkeia server can be obtained with the command:

  ```
  arkc -library -list
  ```

- libid(Optional)
  The libid parameter is the ID assigned to the library when it was created by Arkeia. This ID value can be used instead of the name parameter described above. For instance, the two commands below are equivalent, assuming that 3cb46702 is the correct ID of the Library named Lib04:

  ```
  arkc -library -delete -D name=Lib04
  arkc -library -delete -D libid=3cb46702
  ```

Duplication deletion

The correct syntax to delete a duplication object is the following one:

```
arkc -duplication -delete -[D|I<input file>] <parameters...>
```

The following parameters can be accepted for the deletion of a duplication object:

- name(**Required**)
  The name parameter should contain the name of the duplication object to be deleted. For instance:

  ```
  arkc -duplication -delete -D name=FinanceDup
  ```

- dupid(Optional)
  The dupid parameter is the ID assigned to the duplication object when it was created by Arkeia. This ID value can be used instead of the name parameter described above. For instance, the two commands below are equivalent, assuming that 3ca99702 is the correct ID of the duplication object named FinanceDup:

  ```
  arkc -duplication -delete -D name=FinanceDup
  arkc -duplication -delete -D dupid=3ca99702
  ```

8. The modify command of arkc

The modify command of arkc is used to change the characteristics of an Arkeia object once it has been created. To create an object, please refer to *The create command of arkc* on page *15*.

The modify command has the following general syntax:

```
arkc -[object] -modify -[D|I<input file>] <parameters...>
```

The rest of this chapter will deal with the different objects that can be manipulated using the modify command.

Tape modification

To modify a tape's characteristics, the syntax of `arkc` is the following:

```
arkc -tape -modify -[D|I<input file>] <parameters...>
```

To modify a tape, it is necessary to supply to `arkc` its `name` or its `tpid`. For more information on these parameters, please refer to *The create command of arkc* on page *15*. To obtain a list of the names of the available tapes, it is always possible to enter the following command:

```
arkc -tape -list
```

The following tape parameters can be changed using the `modify` command: `plname`, `plid` (Tape Pool name and ID), `recycle_in`, `recycle_mode` (Recycling functions), `access` (Access rights of the tape), `comment`. These parameters are described in more detail in the chapter *The create command of arkc* on page *15* of this manual.

Drive modification

To modify a tape drive characteristics, the syntax of arkc is the following:

```
arkc -drive -modify -[D|I<input file>] <parameters...>
```

To modify a tape drive, it is necessary to supply to `arkc` its `name` or its `drvid`. For more information on this parameter, please refer to *The create command of arkc* on page *15*. To obtain a list of the names of the available tape drives, it is always possible to enter the following command:

```
arkc -drive -list
```

To obtain a detailed list of the existing characteristics of a specific tape drive (including the `drvid` parameter), it is possible to enter the command:

```
arkc -drive -list -D name=name_of_drive
```

For instance, the following command will return all relevant information on the tape drive named `Sony AIT2`:

```
arkc -drive -list -D name=[Sony AIT2]

PID=19533
DRVID=3bc1d31b
COMMENT=Sony AIT-2 drive plugged on Zeus
NODE=zeus
OWNER=jsmith
DRV_NUM=0
LIBID=0
DRV_DENY=0
STATUS=E
TIME_BEFORE_CLEAN=32767
USAGE_TIME=14924
NB_LOADS=0
CONTROL_DEV=/dev/st0
NONREWIND_DEV=/dev/null
REWIND_DEV=/dev/st0
DRV_TYPE=STD_AIT
NAME=Sony AIT2
TIME_AFTER_CLEAN=14924
```

The following tape drive parameters can be changed using the `modify` command: `rewind_dev` (tape drive UNIX controller), `access` (access rights of the tape drive),

comment. These parameters are described in more detail in chapter 3 of this manual: The create command of arkc.

Savepack modification

To modify a Savepack with arkc, the following syntax should be used:

```
arkc -savepack -modify -[D|I<input file>] <parameters...>
```

Please note that, when modifying a Savepack with arkc, it is necessary to supply the parameter name or the parameter skid (name or ID of the Savepack to be modified).

The following Savepack parameters can be changed through the modify command:

- allowed_fs(file systems allowed)
- retry(number of allowed retries)
- compress(compression scheme to be used for back-up)
- crypt(cryptographic function to be used)
- symb_link(follow-through of symbolic links)
- follow_fs(follow-through of file systems)
- inc_filter(inclusion filter)
- exc_filter(exclusion filters)
- cmd_before, do_cmd_before (pre-backup command)
- cmd_after and do_cmd_after (post-backup commands behavior)
- comment

For more information on these parameters, please refer to *The create command of arkc* on page *15*.

The main use of the modify command, when it applies to Savepack, is to add or remove trees, files and their relevant parameters. The rest of this section will therefore detail the modification of the Arkeia Savepacks through the use of the modify command of arkc.

The information regarding a specific tree have to entered in a configuration file, and each item should be separated by a pair of <ITEM>...</ITEM> statements. For instance, the following lines are valid tree information:

```
<ITEM>
tree_name=station01:/home/jack/mail
</ITEM>

<ITEM>
tree_name=station01:/home/jack/CADfiles
</ITEM>

<ITEM>
tree_name=station01:/home/jack/project/documentation
</ITEM>
```

These lines, inserted in a configuration file, would indicate that, on the computer named station01, the following directories and their contents should be backed up by Arkeia: /home/jack/mail, /home/jack/CADfiles and /home/jack/project/documentation.

The following parameters can be added to the <ITEM>...</ITEM> statements:

- tree_name(Optional)
  The tree_name parameter defines the new tree that will be added to a Savepack. This parameter has the following format:

  name_of_machine:/directory1/directory2/...

  For instance:

  tree_name=station01:/home/jack/project/programming/C_files

  Please note that, to delete a specific tree from within a Savepack, you need to use the parameter dtree_name on the command line. The parameter tree_name and its output FULLNAME (see example above) are equivalent (FULLNAME is the internal information contained within the Arkeia database).

  The examples above are valid for UNIX machines, running operating systems such as Linux or Solaris. For Windows machines, the complete syntax of tree_name is the following:

  tree_name=*machine_name*:*drive_letter*:/directory1/...

  For instance:

  tree_name=dune:c:/common/documents/sales/2000

  Please note the second : after the drive letter. Please note also that arkc requires normal slashes / instead of the standard inverted slashes \ of Windows

Please note: Never use the internal Arkeia name (FULLNAME) whenever you enter a command or edit an arkc input file. Please note as well that the name of a Savepack and the name of a tree are two different parameters. To add a new tree to an existing Savepack, it is therefore necessary to enter either, on the command line:

```
arkc -savepack -modify -D name=station01 tree_name=station01:/home/linda
```

or, within an input file:

```
# Saves Linda home directory on computer station01

<ITEM>
tree_name=station01:/home/linda
</ITEM>
```

Any other syntax than the two shown above will be rejected by arkc. Please note that the two possibilities above are equivalent.

- tree_family(Optional)
  Arkeia uses a sequential procedure to save the trees contained within a Savepack. In order to benefit from Arkeia's advanced parallelism functions, it is necessary to group the different trees in different *families*, using the tree_family parameter. This allows Arkeia to save these tree families at the same time. The values that can be accepted by tree_family are numerical, from 0 to 128. For instance:

  ```
  # Savepack configuration file for station01
  # This contains Jack Smith files

  <ITEM>
  tree_name=station01:/home/jack/mail
  tree_family=1
  </ITEM>

  <ITEM>
  tree_name=station01:/home/jack/CADfiles
  tree_family=0
  </ITEM>

  <ITEM>
  tree_name=station01:/home/jack/project/documentation
  tree_family=0
  </ITEM>
  ```

  The example above defines two tree families: 0 and 1. Between these two families, it is now possible to define the importance of each, and the order in which they are going to be saved. Please see below, the tree_priority and tree_chain parameters for more information.

- tree_priority(Optional)
  The parameter tree_priority defines the priority to be assigned to a particular tree. This parameter accepts numerical values, starting from 0 (highest priority) to 100 (lowest priority). Trees are then backed up according to their priorities. For instance, starting with the example above:

```
# Savepack configuration file for station01
# This contains Jack Smith files

<ITEM>
tree_name=station01:/home/jack/mail
tree_family=1
tree_priority=1
</ITEM>

<ITEM>
tree_name=station01:/home/jack/CADfiles
tree_family=0
tree_priority=0
</ITEM>

<ITEM>
tree_name=station01:/home/jack/project/documentation
tree_family=0
tree_priority=0
</ITEM>
```

  In the example above, the two families 0 and 1 are given different levels of priority. Tree family number 0 (/home/jack/CADfiles and /home/jack/project/documentation) are given a higher priority to reflect their higher importance, while tree 1 is given a lower priority. This gives the result that both families will be backed up in parallel by Arkeia, while giving a higher priority to the documentation and CADfiles directories. The default value of tree_priority is 50.

- tree_chain(Optional)
  The parameter tree_chain is used to create a dependency between trees contained in a Savepack. If two trees are created with the same tree_chain value (from 1 to 128), a dependency is created between the two trees. To control the backup order, the tree_priority should be used (please see above for more information). Please note that a value of 0 means there are no chaining (no dependencies) between trees.

- tree_type(Optional)
  The parameter tree_type defines the type of tree that will be a part of the Savepack. This parameter can accept three different values, which are:

  TREE(standard object file or directory)
  OBJECT(the result of a command should be saved)
  RAW(physical device, such as /dev/fd0)

This parameter determines which other parameters can be applied to the tree. Please see below for more examples. By default, the value of `tree_type` is `TREE`.

The following parameters can be used only if the `tree_type` parameter is set to `TREE`. They are the equivalent, for a single tree, of the general commands available for a compete Savepack. As such, they will not be detailed: please refer to *The create command of arkc* on page *15* for more information on these different parameters.

- `tree_comment`(Optional)
  The parameter `tree_comment` assigns a descriptive information to the tree. For more information on this parameter, please refer to *Savepack creation* on page *22*.

- `tree_retry`(Optional)
  The parameter `tree_retry` assigns a retry value to the tree. For more information, please refer to the `retry` parameter in *Savepack creation* on page *22*.

- `tree_compress`(Optional)
  The parameter `tree_compress` chooses a compression scheme for the tree. For more information, please refer to the `compress` parameter in *Savepack creation* on page *22*.

- `tree_crypt`(Optional)
  The parameter `tree_crypt` chooses a cryptographic scheme for the tree. For more information, please refer to the `crypt` parameter in *Savepack creation* on page *22*.

- `tree_cmd_before`(Optional)
  The parameter `tree_cmd_before` describes a command that should be executed before the tree is backed-up. For more information, please refer to the `cmd_before` parameter in *Savepack creation* on page *22*.

- `tree_do_cmd_before`(Optional)
  The parameter `tree_do_cmd_before` describes the behavior of Arkeia in the case of a failure of the command described in `tree_cmd_before`. For more information, please refer to the `do_cmd_before` parameter in *Savepack creation* on page *22*.

- `tree_cmd_after`(Optional)
  The parameter `tree_cmd_after` assigns a command to be executed after the back-up of the tree by Arkeia. For more information, please refer to the `cmd_after` parameter in *Savepack creation* on page *22*.

- `tree_do_cmd_after`(Optional)
  The parameter `tree_do_cmd_after` describes the behavior of the command entered in `tree_cmd_after` in the case of a failure of the tree backup. For more information, please refer to the `do_cmd_after` parameter in *Savepack creation* on page *22*.

- `tree_allowed_fs`(Optional)
  The parameter `tree_allowed_fs` determines the behavior of the Arkeia backup in case the tree contains NFS links. For more information, please refer to the `allowed_fs` parameter in *Savepack creation* on page *22*.

- `tree_follow_fs`(Optional)
  The parameter `tree_follow_fs` determines if Arkeia should follow shared file systems (for instance: NFS) while saving the tree. For more information, please refer to the `follow_fs` parameter in *Savepack creation* on page *22*.

- `tree_inc_filter`(Optional)
  The parameter `tree_inc_filter` determines an include filter for the tree backup. For more information, please refer to the `inc_filter` parameter in *Savepack creation* on page *22*.

- `tree_exc_filter`(Optional)
  The parameter `tree_inc_filter` determines an exclusion filter for the tree backup. For more information, please refer to the `exc_filter` parameter on page 24.

Drivepack modification

To modify a Drivepack using the `modify` command of `arkc`, the following syntax should be respected:

`arkc -drivepack -modify -D name=`*`name_of_drivepack`* `<`*`parameters`*`>`

It is highly recommended to use an input file (with the option `-I`) to modify a Drivepack. Each item to be added to a Drivepack should be contained within an `<ITEM>...</ITEM>` statement. The parameters that can be added to such an input file are the following ones:

- `drvname`(Optional)
  The parameter `drvname` contains the name of the drive to be added to the

Drivepack. You can obtain a list of drives installed on the Arkeia server by entering the following command:

```
arkc -drive -list
```

To add a drive to a Drivepack, the following syntax should therefore be used:

```
<ITEM>
drvname=DLT01
</ITEM>
```

The above `<ITEM>` will add the drive `DLT01` to a Drivepack, assuming a drive of that name exists on the Arkeia server.

- `drvid`(Optional)
  The parameter `drvid` contains the ID of the drive to be added to the Drivepack. This parameter is used in much the same way than the parameter `drvname` above. For instance, the following `<ITEM>` statement will add the drive `DLT01` to a Drivepack, assuming `3bc419c3` is the correct ID of this drive:

  ```
  <ITEM>
  drvid=3bc419c3
  </ITEM>
  ```

- `priority`(Optional)
  The parameter `priority` defines the priority level assigned to the drive. This parameter can accept values from `1` to `128`, with the default (and highest possible) value being `1`. Please note that this parameter can only be used within an input file, and not on the command line.

The following parameters can also be modified using the `-modify` command of `arkc`: `name`, `dkid` (Drivepack name and ID) and `nbdrive` (number of drives to be used in the Drivepack). Since these are functionally equivalent to the parameters of the create command, please refer to *The create command of arkc* on page *15* for more information.

User modification

To modify an Arkeia user, the parameters to be used are the same as with the `create` command. For more information, please refer to *The create command of arkc* on page *15*. The valid syntax is therefore:

`arkc -user -modify -[D|I] <`***parameters***`>`

The parameters used to modify a user are `name`, `role`, `node`, `email`, `passwd` and `vpasswd`. For more information on these, please refer to *The create command of arkc* on page *15*.

Library modification

To modify an existing library, please refer to the library creation described in *Library creation* on page *29* of this manual. Please note that the correct syntax is the following:

```
arkc -library -modify -[D|I<input file>] <parameters...>
```

While modifying a library, only the following parameters can be used: `name`, `libid` (Library name and ID), `libdev` (control device for the library) and `comment`. For more information on these parameters, please refer to *The create command of arkc* on page *15*.

Duplication modification

To modify an existing tape duplication object, the correct syntax is:

```
arkc -duplication -modify -[D|I<input file>] <parameters...>
```

The parameter required to modify a duplication object is the following one:

- `name`(**Required**)
  The `name` parameter is assigned to the duplication object during its creation. To modify an object, its name is required by `arkc`. For instance:

  ```
  name=CAD_Dup
  ```

  Please note that this parameter can be replaced by the parameter `dupid`, which contains the ID of the duplication object to be modified.

The optional parameters that can be modified by the `modify` command are the followings:

- `src_tape_name`/`src_tape_id`Source tape name or ID
- `src_drv_name`/`src_drv_id`Source drive name or ID
- `dst_drv_name`/`dst_drv_id`Destination drive name or ID
- `dst_lib_name`/`dst_lib_id`Destination library name or ID
- `dst_slot`Destination slot in library
- `comment`Informational comment

For more information on these parameters and the values they can accept, please refer to *The create command of arkc* on page *15*.

9. The informational commands and objects of arkc

The following `arkc` commands and objects are informational in nature, designed to offer the user detailed (or terse) data about Arkeia objects and operations:

- `list`Displays existing objects.
- `drvlist`Displays existing drives.
- `type`Displays drives and tapes that Arkeia can manage.
- `role`Displays users roles and attributions.
- `done`Displays completed backups.
- `running`Displays backups or restorations in progress.
- `status`Displays progress information on running jobs.
- `statistics`Displays tape statistics information.
- `debug`Displays miscellaneous information.
- `journal`Displays information from Arkeia log.

This chapter will detail all these objects and command, their usage, and the helpful information they can supply to Arkeia users.

Listing objects with the list command

The standard syntax for the `list` command is the following one:

```
arkc -object -list -[D|I] <parameters>
```

This command will display all the *`objects`* currently available on the Arkeia server. For instance, the following command will display a list of all the tapes available on the Arkeia server:

```
$ arkc -tape -list

name=DLT_Tape_1
name=DLT_Tape_2
name=DLT_Tape_3
name=DLT_Tape_4
name=DLT_Tape_5
name=Test_Tape_01
```

If more information is needed on a specific object, the `list` command is able to display all relevant information, provided the name of the object is supplied as a parameter. Based on the example above, here is the way to display all the information on `Test_Tape_01` which is contained in the Arkeia database:

```
$ arkc -tape -list -D name=Test_Tape_01

REMAIN_SPACE=50
TOTAL_SPACE=0
REMAIN_USE=0
CID=3a7834d7
TPID=3a7834d7
PLID=3a78354e
RECYCLE_MODE=0
RECYCLE_IN=0
VOLTAG=/home/jack/files/test/arkeia/backup.tmp
COMMENT=This is a test tape.
USED_SPACE=0
CREATION_DATE=980956375
OWNER=root
SEG_END=0
ST_WORN=0
```

```
ST_ACCESS=TPSTA_MANUAL
ST_CONTENT=TPSTC_NEW
ST_DENY=16
TP_TYPE=FILE 50MB
NAME=Test_Tape_01
```

The `list` command can be applied to the following Arkeia objects:

- `tape`
- `pool`
- `drive`
- `savepack`
- `drivepack`
- `user`
- `library`
- `duplication`

In addition, the command `list` accepts the following filters:

- Filters for the `drive` object:
  The following filters are available for the `drive` object: `name` (to list the names of the drives) and `drvid` (to list the ID of the drives). It is possible to combine the two, with, for instance:

  ```
  arkc -drive -list -Fname -Fdrvid
  ```

  The command above would list all the drives available on a server, by putting the name of the drive first, followed by its ID. Please note that there is no space between `-F` and the name of the filter.

- Filters for the `savepack` object:
  The following filters are available for the `savepack` object: `name` (lists only the name of the savepacks) and `skid` (lists only the ID of the savepacks). Please see above for an example of combined filters.

- Filters for the `drivepack` object:
  The following filters are available for the `drivepack` object: `name` (to display only the name of the drivepacks) and `dkid` (to display only the IDs of the drivepacks). Please see above for an example of combined filters.

- Filters for the `library` object:
  The following filters are available for the `library` object: `name` (displays only the name of the library) and `libid` (displays only the ID of the library). Please see above for an example of combined filters.

Listing files present on a tape with the list command

The object `file` is used to obtain a list of files contained on a given tape. When using the command `arkc -file -list`, it is therefore necessary to supply the name of the tape to be listed with the `tpname` parameter. The standard syntax is therefore:

```
arkc -file -list -D tpname=name_of_tape
```

For instance:

```
arkc -file -list -D tpname=DLT_140
```

The command above will list all the files contained on the tape named `DLT_140`. For more information on the `file` object, please refer to the second part of this manual.

It is possible to filter the output of -file -list command with the following filter:

- `file`Displays only the files present on the tape.

Listing the drives of a library with the drvlist command

For the `library` object, arkc offers an additional `list` command available, which is the `drvlist` command.

While the `list` command, when applied to the `library` object, will list all the available libraries connected to a Arkeia server, the `drvlist` command will list the drives that are installed in a specific library. The syntax of this command is the following:

```
arkc -library -drvlist -[D|I] <parameters...>
```

The parameters that can be applied to this command are the following ones:

- `name`(**Required**)
  The `name` parameter contains the name that was assigned to the library when it was created. Please note that this parameter is required by the `drvlist` command. For example:

  ```
  arkc -library -drvlist -D name=ArkeiaLib01
  ```

  The above command will list all the tape drives installed within the library named `ArkeiaLib01`.

- `libid`(Optional)
  The `libid` parameter is the ID that was assigned to the library when it was created. It can replace the name parameter. For instance, the two commands below are equivalent, assuming that `3b614c30` is the valid ID for library `Autopack30`:

  ```
  arkc -library -drvlist -D name=Autopack30
  arkc -library -drvlist -D libid=3b614c30
  ```

The output of the drvlist command can be modified with the help of the following filters:

- drvnameDisplays only the names of the drives present in a library.
- drvidDisplays only the IDs of the drives present in a library.
- drvnumDisplays only the numbers of the drives present in a library.

Please note it is possible to combine several filters. For instance, the following command would list the drives present in library IBMlib and display both their names and IDs:

```
arkc -library -drvlist -D name=IBMlib -Fdrvname -Fdrvid
```

Obtaining information on objects with the type command

The type command is designed to provide information on the type of objects that can, for instance, be created by arkc. Its syntax is as simple as the one of the list command:

```
arkc -object -type
```

We have seen several examples in the previous sections of this manual. Here is a complete list of the information that can be obtained with the type command:

- tape
  The command arkc -tape -type will list all the tape standards that can be managed by Arkeia. A complete list can be found in chapter 3, the create command, in the section titled tape creation.

- drive
  The command arkc -drive -type will list all the drive standards that can be managed by Arkeia. A complete list can be found in chapter 3, the create command, in the section titled drive creation.

- library
  The command arkc -library -type will list all the libraries that can be managed by Arkeia. A complete list can be found in chapter 3, the create command, in the section titled library creation.

The role command

The role command is a limited version of the type command, that applies only to the user objects of Arkeia. To obtain all the roles available to Arkeia users, enter the command:

```
arkc -user -role
```

The three roles that are available for Arkeia users are: OPERATOR, USER and ADMINISTRATOR. Please note that, when a user is created, a role should be defined and

entered in upper case, under the parameter role. For instance, the following input file defines a valid Arkeia user:

```
# ----- File name jack.txt -----
# User creation file for arkc.
# ----------------------------
# Basic features

name=[Jack Smith]
role=ADMINISTRATOR

node=station01

# Optional features
email=jack@foobar.com

# End of file.
```

To create the user Jack Smith, the following command should therefore be entered, assuming the input file is named jack.txt:

```
arkc -user -create -Ijack.txt
```

Displaying the backups completed with the done command

It is possible to display a list of all backup operations completed by using the done command of arkc.

The syntax of this command is the following one:

```
arkc -backup -done [-[D|I<input file>] <parameters...>]
```

The command arkc -backup -done will simply list all the backup that have been completed. To obtain a more informational message, it is necessary to filter the output of this command with the following filters:

- bksidDisplays only the ID of the backups.
- dateDisplays only the date (numerical) of the backups.
- sdateDisplays only the date (string) of the backups.

For instance, here are a few examples of filters, and their outputs:

```
$ arkc -backup -done
bksid=3a6718dc

$ arkc -backup -done -Fsdate
sdate=2001/01/18 16:25

$ arkc -backup -done -Fsdate -Fbksid
bksid=3a6718dc
sdate=2001/01/18 16:25
```

Displaying the list of jobs in progress with running

The running command can be used to display the list of backups or restoration that are in progress on the Arkeia server. Further information about a specific job can then be queried using the status command (see below).

The standard syntax for the running command is the following:

```
arkc -[backup|restore] -running
```

For instance, to display a list of all backup operations currently in progress, it is possible to enter:

```
arkc -backup -running
```

This command will then display the IDs of all the backup in progress on the Arkeia server. To obtain more information on a specific job, please refer to the status command below.

Displaying information on a job in progress with status

The status command can be used to display information on either backups or on restorations. This will detailed in the next two sections:

Status of a running backup

To obtain detailed information on a running backup, the status command should be used. This command has the following syntax:

```
arkc -backup -status -[D|I<input file>] <parameters...>
```

This command accepts only one parameter, which is detailed below:

- bksid**(Required)**
  The bksid parameter contains the ID of the backup whose detailed status should be displayed. This ID is displayed when a backup is started with the start command. For more information, please refer to the next chapter of this manual: *The start and the stop commands of arkc* on page *63*.

For instance, the following command:

```
arkc -backup -status -D bksid=3bcc2793
```

will return the possible values:

- 0The backup job is running.
- 1The backup needs a new tape.
- 2The backup is now finished.

If the value returned is 1, the status command also returns two values, which are:

- drvnameName of the drive to be used for the new tape.
- tpnameName of the new tape to be inserted in the drive.

Please note that it is possible to modify the behavior of a running backup by using the connect command to modify its parameters. For more information on the connect command, please refer to *Modifying a job with the connect command* on page *73*.

Status of a running restoration

To obtain detailed information on a running restoration, the status command should be used. This command has the following syntax:

```
arkc -restore -status -[D|I<input file>] <parameters...>
```
This command accepts only one parameter, which is detailed below:

- `rstid`**(Required)**
  The `rstid` parameter contains the ID of the restoration whose detailed status should be displayed. This ID is displayed when a restoration is started with the `start` command. For more information, please refer to the *The start and the stop commands of arkc* on page *63*.

For instance, the following command:

```
arkc -restore -status -D rstid=3bcc4545
```
will return the possible values:

- `0`The restoration is in progress.
- `1`The restoration needs a new tape.
- `2`The restoration is now completed.
- `3`The restoration requires a new drive.

If the value returned is `1`, the `status` command also displays two values, which are:

- `drvname`Name of the drive to be used for the new tape.
- `tpname`Name of the new tape to be inserted in the drive.

If the value returned is `3`, the only value displayed will be `drvname`, as the restoration only requires a new drive, and not a new drive and a new tape.

Please note that it is possible to modify the behavior of a restoration in progress by using the `connect` command to modify its parameters. For more information on the `connect` command and its uses, please refer to *Modifying a job with the connect command* on page *73*.

Displaying additional tape information with statistics

It is possible to obtain detailed statistics on a given tape by using the `statistics` command of `arkc`. This command is especially useful when the time comes to retire a tape at the end of its life and delete it from the Arkeia database.

The syntax for the `statistics` command is the following one:

```
arkc -tape -statistics -[D|I<input file>] <parameters...>
```
The possible parameters of the `statistics` command are the following:

- `name/tpid`**(Required)**
  The `name` and `tpid` parameters contain, respectively, the name and the ID of the tape whose statistics should be displayed. Please note that only one parameter is needed. For instance, the following commands are equivalent, assuming `3b98c4b3` is the correct ID for the tape named `DLT00`:

  ```
  arkc -tape -statistics -D name=DLT00
  arkc -tape -statistics -D tpid=3b98c4b3
  ```

- plname/plid(Optional)
  The plname and plid parameters contain, respectively, the name and the ID of the tape pool to be analyzed by the statistics command. Please note that only one parameter is needed. If a tape pool is used, all the tapes it contains are analyzed and their statistics displayed. For instance, the following commands would display the statistics of the tape pool named VXA, whose ID is 3aaf90a4:

  ```
  arkc -tape -statistics -D plname=VXA
  arkc -tape -statistics -D plid=3aaf90a4
  ```

Checking parameters and errors with debug

The debug object has several different uses which may be of interest. The following commands can be applied to debug:

- -debug -whoDisplays information on the Arkeia server/login.
- -debug -inputDisplays a detailed analysis of an input file.
- -debug -errorsDisplays detailed information on error messages.

The different uses of debug are detailed below.

Displaying useful information on the Arkeia server/login in use.

The first use of the debug object is to display a terse summary of the information related to the Arkeia server and the login currently in use. This command does not accept any parameters. Its syntax is the following:

```
arkc -debug -who
```

A typical output of -debug -who could be the following:

```
$ arkc -debug -who


Preference file : /usr/knox/arkc/arkc.param
Backup server   : stormbringer
Server version  : 4.2.8-1
Login           : root
```

In the example above, the preference file used by arkc is /usr/knox/arkc/arkc.param, the Arkeia backup server is named stormbringer, its version is 4.2.8-1 and the login used is root.

Checking the content of an input file

Another use of debug, with the input command, is to quickly parse an input file and return any and all errors found in this input file. The correct syntax of this command is the following:

```
arkc -debug -input -I<input file>
```

For instance, the following command would check the file named: create_user.txt:

```
$ arkc -debug -input -Icreate_user.txt
```

```
name=Jack Hill
role=ADMINISTRATOR
node=stormbringer
email=jack.hill@foobar.com
```
The output of the command should be self-explanatory. For more information on the creation of a user, please refer to *The create command of arkc* on page *15*.

From the example above, it is easy to understand that the only parameter accepted by the command `-debug -input` is the input file to be analyzed.

Displaying error codes and their meanings

The final use of the `debug` object is to display a list of all `arkc` error codes, along with their meanings. To display error codes and meanings, it is necessary to use the `errors` command.

The syntax for this command is the following:

```
arkc -debug -errors
```
This command does not accept any parameters. On some systems, it may be useful to redirect the result of this command into a file and to save (or print) this file for future reference, as illustrated below:

```
arkc -debug -errors > arkc_error_list.txt
```
The command above will create a short text-only file, containing all the error codes of `arkc`.

A short list of `arkc` error and exit codes

The following is a short list of all the error codes that can be encountered while using `arkc`. Please note that this list is not the full output of the `-debug -errors` command, but rather a list of the most frequent errors. For each error code, a short explanation is also given.

Main exit codes

Request succeeded	0
Internal error	11
Internal error	12
Error while loading command line options	13
Error while checking command line options	14
Error loading preference file	15
Internal error	16
Internal error	17
Connection error	18
License error	19
Internal error	20
Error while loading command parameters	21

Main exit codes

Error while executing request	22
Error while displaying command result	23
Internal error	24
Error: wrong server version	25

Exit codes returned by the `-drive -read` command:

Drive is full	0
Drive is empty	1

Error messages string

```
MLC_ERROR_DEFAULT
Internal error

MLC_ERROR_ALLOCATION
Internal error

MLC_ERROR_BAD_SERVER_VERSION
You tried to use arkc with an Arkeia Backup Server whose version
is older than version 4.1.8

MLC_ERROR_INVALID_PARAMETER_X
Internal error

MLC_ERROR_ARGUMENT_NOT_FIND
You haven't set a parameter or an option with a proper value.
For example: -property without a parameter name.

MLC_ERROR_INVALID_OBJECT
The current command is waiting for an object value.
Your command may contain two different objects.

MLC_ERROR_INVALID_ACTION
You tried to execute an operation which is not allowed
with the object chosen.

MLC_ERROR_LOADING_OBJECT
The arkc command line contains a command but not an object.

MLC_ERROR_LOADING_ACTION
A command could not be found on the command line.

MLC_ERROR_INPUTFILE_ARGUMENT
An input file option (-I) has been used, without specifying
a file name. Do not insert a space character between '-I'
and the filename.

MLC_ERROR_OUTPUTFILE_ARGUMENT
An output file option (-O) has been used, without specifying
a file name. Do not insert a space character between '-O'
and the filename.
```

```
MLC_ERROR_ARGUMENT_SYNTAX
arkc has detected a syntax error while parsing the parameters
list. Perhaps a parameter value or a '=' character has not been
entered.

MLC_ERROR_ARGVALUE_MISSED
arkc has detected a syntax error while parsing the parameters
list. Perhaps a parameter value or a '=' character has not been
entered.

MLC_ERROR_ARGUMENT_MISSED
arkc has detected a syntax error while parsing the parameters
list. Perhaps a parameter value or a '=' character has not been
entered.

MLC_ERROR_TO_MUCH_BRACKETS
arkc has detected a syntax error while parsing the parameters
list. It may be possible that there are too many characters
'[[' or ']]' present on the command line.

MLC_ERROR_TO_MUCH_AFFECTATION
arkc has detected a syntax error while parsing the parameters
list. It may be possible that there are too many characters
'==' on the command line.

MLC_ERROR_ARGUMENT_NOT_USED
Internal error

MLC_ERROR_UNKNOWN_ARGUMENT
arkc has detected a syntax error on the command line. There
may be a bad argument. Please try the '-helpme' option.
(see FAQ, help section)

MLC_ERROR_INPUTFILE_OPEN
arkc could not open the file specified as the input file
(with the '-I' option) on the command line.

MLC_ERROR_INPUTFILE_NOT_EXIST
arkc could not find the file specified as the input file
(with the '-I' option) on the command line.

MLC_ERROR_INPUTFILE_PARSING
arkc could not find the file specified as the input file
(with the '-I' option) on the command line. You can try
to use '-debug' to analyze your input file.
For instance: # arkc -debug -input -Iinputfile

MLC_ERROR_INPUTFILE_CLOSE
arkc could not close the file specified as the input
file (with the '-I' option) on the command line.

MLC_ERROR_INPUTFILE_NEEDED
An input file is needed.

MLC_ERROR_INPUTSTD
arkc could not parse the standard input (with the '-' option).

MLC_ERROR_OUTPUT_WRITE
arkc could not write to the output file specified
(wit the '-O' option).

MLC_ERROR_OUTPUTFILE_OPEN
arkc could not open the file specified as the output
```

```
file (with the '-O' option) in the command line.

MLC_ERROR_OUTPUTFILE_WRITING
arkc could not wtite to the output file specified
(with the '-O' option) on the command line.

MLC_ERROR_OUTPUTFILE_CLOSE
arkc could not close the file specified as the output
file (with the '-O' option) on the command line.

MLC_ERROR_FUNCTION_AFFECTATION
Internal error.

MLC_ERROR_INIT_PREFS
arkc could not use the preference file (see FAQ)

MLC_ERROR_CONNECTION
arkc could not establish a connection to the backup
server. Please refer to the arkc FAQ for more information
on the possible connection problems.

MLC_ERROR_EXECUTING_FUNCTION
arkc could not execute a request. Please note that
the arkc configuration and command syntax are correct.

MLC_ERROR_ARGUMENT_FUNCTION
Internal error

MLC_ERROR_RETURNING_FUNCTION
Internal error

MLC_ERROR_BAD_VALUE
arkc has found an incorrect parameter value on the command line.

MLC_ERROR_LOADING_HELP_FILE
Internal error

MLC_ERROR_LOADING_INDEX_FILE
Internal error

MLC_ERROR_UNKNOWN_INDEX
Internal error

MLC_ERROR_PARSING_HELP_FILE
Internal error
```

Displaying log information with the journal object

The `arkc journal` object is used to display the log of the Arkeia server. The correct syntax to use for this is the following:

`arkc -journal -`***command*** `-D <`***parameters***`...>`

The different commands that can be used with the `journal` object are the following:

- `-journal -all`Displays all log information.
- `-journal -jtape`Displays only tape-related information.
- `-journal -jdrive`Displays only drive-related information.
- `-journal -jbackup`Displays only backup-related information.

- `-journal -jrestore`Displays only restoration-related information.

The rest of this section will detail the different commands listed above. Since all the commands accept the same parameters, these will be detailed at the end of the section.

Displaying the complete log information

The correct syntax to display the complete information contained in the Arkeia log is the following:

```
arkc -journal -all -D <parameters...>
```

For instance, the following command will display the complete Arkeia logs:

```
$ arkc -journal -all


...
2000/11/06 17:46:46 I090001 S0 Tape 'X4': created by 'root'
2000/11/19 20:00:43 I090001 S1005065323 Tape 'X4': removed by 'root'
...
```

In the example messages above, we can see that the user `root` has created a tape named `X4` and deleted it 13 days later. A normal log would, of course, contain a lot more information.

For more information on the parameters that can be accepted by this command, please refer to the section named *Possible parameters for the journal object and commands* on page *61*.

Displaying the tape-related information

The syntax to display only the tape-related information contained in the Arkeia log is the following:

```
arkc -journal -jtape [-D] <parameters...>
```

For more information about the parameters that can be accepted by this command, please refer to the section named *Possible parameters for the journal object and commands* on page *61*.

Displaying the drive-related information

The syntax to display only drive-related information from the Arkeia log is the following:

```
arkc -journal -jdrive [-D] <parameters...>
```

Displaying the restoration-related information

The syntax to display only restoration-related information from the Arkeia log is the following:

```
arkc -journal -jrestore [-D] <parameters...>
```

Possible parameters for the journal object and commands

The parameters that can be accepted by the `journal` object are detailed below. Please note that these parameters apply to all the commands that were detailed above (`all`, `jtape`, `jdrive`, and `jrestore`), and that they are all optional.

The only exception to this rule is the jbackup command, which has one parameter (bksid) that is not required by all the other commands. This command will be detailed after the common parameters that apply to all the journal commands.

- month(Optional)
  The parameter month limits the display of the journal object to a specific month. The value of month can go from 1 (January) to 12 (December). For instance:

  ```
  arkc -journal -jbackup -D month=10
  ```

  The command above will display all the backups that took place during the month of October of the current year. By default, the month displayed by the journal object is the current month.

- level(Optional)
  The parameter level specifies the quantity of information that should be displayed by a command. This parameter can accept three different values: %I (information messages), %W (warning messages) and %E (error messages). The default value for this parameter is %I%W%E, which displays all the log messages (information, warnings and errors). For instance:

  ```
  arkc -journal -jrestore -D level=%W%E
  ```

  The command above will only display the warnings and errors that were reported by the Arkeia server during the restoration operations. On the other hand, the information messages won't be displayed. The level parameter therefore limits the information to the most interesting messages, which can be quite useful to track the source of a problem.

- filter(Optional)
  The filter parameter adds or removes specific data from the information returned by the Arkeia log. This parameter accepts the following values: %D (date), %T (time), %L (level, see above), %I (information code) and %S (message string). The default value for the filter parameter is %D%T%L%I%S, meaning that all information returned by the log is displayed. For instance:

  ```
  arkc -journal -jrestore -D filter=%I%S
  ```

  The command above would limit the information returned by the Arkeia log to the information code, followed by the message string. For instance, here is a typical output of such a command:

```
$ arkc -journal -jrestore -D filter=%L%S

W060006 Start of restoration
```

```
I060016 wks_dtp:/home/tina/Documents/Tips_ML04.txt
I060021 Loading tape 'DLT02' in drive 'TinaDLT'
I060008 [1] Errors: 0
I060010 [1] Directories not treated: 0
I060011 [1] Components not treated: 0
I060020 Unloading tape from drive 'TinaDLT'
W060020 End of restoration
```

Displaying the backup-related information

The syntax to display only backup-related information from the Arkeia log is the following:

```
arkc -journal -jbackup [-D] <parameters...>
```

The command `jbackup` accepts all the parameters detailed above (`month`, `level` and `filter`) and adds the following required parameter:

- `bksid`(**Required**)
  Using the parameter `bksid`, which contains the ID of a completed backup, will display only the information related to this backup. For instance, you will find below a typical `-journal -jbackup` command and its output (please note the usage of the `filter` parameter):

```
arkc -journal -jbackup -D bksid=3b5316a0 filter=%L%S
```

```
W080006 Start of backup: Savepack "dragon", Drivepack "Ecrix Autopak", Pool "VXA"
W080006 Owner is "root"
I080003 Adding savepack "dragon"
I080010 Loading tape 'VXA_03' in drive 'ECRIX_2'
W080006 Job has been aborted by 'root'
```

10. The start and the stop commands of arkc

The `start` and the `stop` commands of arkc are used only for the following Arkeia objects:

- `backup` (`start` only)
- `restore` (`start` only)
- `duplication`(`start` only)
- `library` (`start` and `stop`)

The rest of this chapter will detail the different possibilities offered by these commands.

Starting a backup

The `start` command can be used to launch a backup with `arkc`. The standard syntax for this command is the following one:

`arkc -backup -start -[D|I] <parameters...>`
The following parameters can be applied to this command:

- `skname`(**Required**)
  The `skname` parameter contains the name of the Savepack to be saved by the backup. This name is assigned to the Savepack during its creation. For instance:

  ```
  arkc -backup -start -D skname=JackHome
  ```

  The command above would start the backup of the Savepack named `JackHome`.

- `plname`(**Required**)
  The `plname` parameter contains the name of the tape pool to be used by the backup. This name is assigned to the tape pool during its creation. For instance:

  ```
  arkc -backup -start -D skname=JackHome plname=Pool01
  ```

  The command above would start the backup of the Savepack named `JackHome`, on the tape pool named: `Pool01`.

- `dkname`(**Required**)
  The `dkname` parameter contains the name of the Drivepack to be used for the backup. This name is assigned to the Drivepack during its creation. For instance:

  ```
  arkc -backup -start -D skname=JackHome plname=Pool01
  dkname=DSK_01
  ```

  The command above would start the backup of `JackHome`, on the tape pool `Pool01` and on the Drivepack named `DSK_01`. Please note that these options should all be contained on the same line.

Please note: The parameters `skname`, `plname` and `dkname`, while they must be entered on the command line to start a backup can be replaced by the parameters `skid`, `plid` and `dkid`. These parameters are respectively the IDs for the Savepack, the tape pool and the Drivepack to be used.

The following optional parameters can be used to start a backup with `arkc`:

- `policy`(Optional)
  The parameter `policy` determines the tape policy to be used for the backup. This parameter can accept two values: `COMPLETE` or `NEW`. When using the `COMPLETE` value, Arkeia will always complete (fill up) the tape which has been

used. If the NEW value is used, Arkeia will always use a new tape when starting a backup. By default, policy has the value COMPLETE.

- wait(Optional)
  The parameter wait determines the behavior of arkc once a backup is launched with arkc -backup -start. This parameter can accept two values: YES or NO. If the value YES is used, arkc will wait until the end of the backup to return, which means the user will not be able to enter any new command until the completion of the backup. If the value NO is used, arkc will return immediately and allow the user to enter other commands.

- email(Optional)
  The parameter email determines if Arkeia sends an email once the backup is finished, or if the backup requires a new tape to be completed. This parameter can accept two values YES and NO. If the value YES is used, Arkeia will send an email to the user, indicating if the backup is finished and/or if the software requires an additional tape to complete the backup operation. If the value NO is used, then Arkeia will not send an email. The default value of email is YES.

Please note: In order to receive an email, the parameter email, defined for a user, must contain a valid email address. Even if the email parameter of a backup is defined, it will not send any information if the user email parameter does not contain any information.

- mode(Optional)
  The parameter mode controls the operational mode of Arkeia for a backup. Two modes (and values of mode) are available: CONTINUOUS and STANDARD. When the STANDARD mode of operation is chosen, the backup will stop once it is completed, and will be considered as terminated by the Arkeia server. When the CONTINUOUS mode is selected, the backup task will not be considered as finished once it has saved all the files in the Savepack, and more Savepacks can be added to the backup task. By default, the value of mode is STANDARD: if you are unsure which mode is best suited to your need, please use the default.

- comment(Optional)
  The parameter comment is used to add a more descriptive comment to a backup. For instance:

  ```
  comment=[Complete user data backup, including email dir]
  ```

  Please note that, as seen above, comments that contain spaces should be enclosed within square brackets [...].

- `retention`(Optional)
  The parameter `retention` determines the duration of the data retention of the backup. This parameter must be completed by the `retunit` parameter (see below) and can accept values starting at `1`. For instance:

  ```
  retention=6
  ```

- `retunit`(Optional)
  The parameter `retunit` determines the type of unit used by the `retention` parameter. It can accept the following values: `DAY`, `WEEK`, `MONTH` and `YEAR`. For instance, to take back the example above:

  ```
  retention=6
  retunit=MONTH
  ```

  This defines a retention period of 6 months for the data of the backup being started with `arkc`.

- `parallelism`(Optional)
  The parameter `parallelism` determines the number of concurrent data flows that will be used for the backup. This parameter can accept numerical values from `1` to `128`. By default, Arkeia uses the maximum number of flows available for its backups.

- `type`(Optional)
  The parameter `type` determines the type of backup that will be performed by Arkeia. This parameter can accept three values: `TOTAL`, `INCREMENTAL` and `ARCHIVE`. By default, the value of `type` is `TOTAL`. For more information about the different type of backup available under Arkeia, please refer to the Arkeia User Manual.

- `based_on_bksid`(Optional)
  The `based_on_bksid` parameter is valid only if the `type` parameter is set to `INCREMENTAL`. In this case, the `based_on_bksid` parameter should contain the ID of the previous backup the new backup is based on.

- `based_on_date`(Optional)
  The `based_on_date` parameter is valid only if the `type` parameter is set to `INCREMENTAL`. In this case, the `based_on_date` parameter should contain the date of the previous backup the new backup is based on.

Starting a restoration

The command `start` allows you to start a restoration operation from the command line. The standard syntax for this command is:

```
arkc -restore -start -[D|I<input file>] <parameters...>
```

The parameters that can be accepted by `arkc` with this command are the following ones:

- `file`(**Required**)
  The `file` parameter must be inserted on the command line or in an input file. This parameter contains the name of the file (or directory) that should be restored. For instance:

  ```
  arkc -restore -start -D
  file=stormbringer:/home/jack/docs/AnnualReport_2000.ps
  ```

  The `arkc` command above, if entered on a command line, will start the restoration of the file named `AnnualReport_2000.ps`, which can be found on the machine named `stormbringer` in directory: `/home/jack/docs/`. Please note that the command above should be entered on a single line.

- `from/to`(Optional)
  The parameter `from` allows the restoration to take place with a redirection. This means that a file restored from a directory `X` can be restored to another directory `Y`. For instance, taking back the example above:

  ```
  arkc -restore -start -Ireports.txt
  ```

  (Content of `reports.txt`:)

  ```
  # Example of restoration __________

  file=AnnualReport_2000.ps
  from=babylon:/home/jack/files/Reports/
  to=babylon:/home/jack/tmp/arkeia/temporary/
  ```

  In the example above the file `AnnualReport_2000.ps`, whose original directory was `/home/jack/files/Reports` on the workstation `babylon`, (`from=` line) will be restored instead to the directory: `/home/jack/tmp/arkeia/temporary` (`to=` line). This allows the user to check the content of the restored file before moving it back to its original position.

Please note: A `from` parameter should always be followed and completed with a `to` parameter. If this is not the case, the restoration will be rejected by `arkc` with an error message.

- `wait`(Optional)
  The parameter `wait` determines the behavior of `arkc` once a restoration is launched with `arkc -restore -start`. This parameter can accept two values: `YES` or `NO`. If the value `YES` is used, `arkc` will wait until the end of the restoration to return, which means the user will not be able to enter any new command until the completion of the operation. If the value `NO` is used, `arkc` will return immediately and allow the user to enter other commands on the command line.

- `drvname`(Optional)
  The parameter `drvname` contains the name of the drive to be used for the restoration, in a single-drive environment. This name is assigned during the creation of the drive. For instance:

  ```
  drvname=SonyDLT
  ```

  On a single-drive system, the parameter above indicates that the drive named `SonyDLT` should be used for the restoration operation.

Please note: It is possible to replace the `drvname` parameter by the `drvid` parameter. The `drvid` should then contain the ID of the drive to be used for the restoration, in the case of a single-drive configuration. This ID is assigned to the drive during its creation and can be obtained by entering the following command:

```
arkc -drive -list -D name=name_of_the_drive
```

- `rstopt`(Optional)
  The `rstopt` parameter determines the different options that should be applied to the files that will be managed by the restoration. This parameter accepts the following values, that will influence the restoration:

  `REST_TIMES`Restore the time information of the files.
  `RST_PERMS`Restore the access rights of the files.
  `RST_BYID`Restore by user identifier.
  `RST_BYNAME`Restore by user name.
  `RST_CHKTIME`Check if the existing file (if any) is newer.

Starting a tape duplication operation

Like other `arkc` objects, a tape duplication operation must be first created (Please refer to *The create command of arkc* on page *15* for more information) and can then be started. To this the proper syntax is the following:

```
arkc -duplication -start -[D|I<input file>] <parameters...>
```

The following parameters are valid when starting a tape duplication operation:

- `name`(**Required**)
  The `name` parameter contains the name of the duplication object that should be started. This name is assigned to the duplication object during its creation. For instance:

  ```
  arkc -duplication -start -D name=[TCD Duplication]
  ```

  The command above would start the tape duplication named `TCD Duplication`.

- `dupid`(Optional)
  The `dupid` parameter contains the ID of the duplication object that should be started. Please note that this parameter can be used instead of the `name` parameter described above. The ID of the duplication object is assigned to it when it is created and can be obtained by entering the command:

  ```
  arkc -duplication -list -D name=<duplication_name>
  ```

  For instance, the two commands below are equivalent, assuming `3b98c4b1` is the correct ID for the tape duplication named `TCD Duplication`:

  ```
  arkc -duplication -start -D name=[TCD Duplication]
  arkc -duplication -start -D dpid=3b98c4b1
  ```

- `force`(Optional)
  The `force` parameter will force the duplication even if the copy and the original tape appear to be identical. The `force` parameter accepts only value `1`, which forces Arkeia to duplicate the tapes, which can be interesting if it is necessary to copy a tape several times. For more information on tape synchronization, please refer to the section named *Tape duplication creation* on page *31* of this manual.

Starting and stopping a library

The final application of the `start` and `stop` commands are related to the tape libraries, which can be managed with the help of these two commands.

Starting a library

In order to use a given library, Arkeia has to start a specialized module, that allows it to *drive* (meaning: use all the functions of) a library. This module is usually launched automatically, for instance when a backup is started. However, it is sometimes necessary to start a library manually. To do this, the following syntax should be respected:

```
arkc -library -start -[D|I<input file>] <parameters...>
```

This command accepts only one parameter, which is:

- name(**Required**)
  The name parameter contains the name of the tape library to be started. This name is assigned to the tape library during its creation. For instance:

  ```
  arkc -library -start -D name=[HP SureStore 1/9 DLT]
  ```

  The command above would start the tape library named HP SureStore 1/9 DLT. Please note the use of the square brackets around the name of the library.

Please note: it is possible to replace the name parameter by the libid parameter. The libid parameter is the ID of the library that should be started. It can be obtained by entering the command:

```
arkc -library -list -D name=name_of_the_library | more
```

Therefore, the following start command is correct, assuming 3af012b1 is the correct ID for the library HP SureStore 1/9 DLT:

```
arkc -library -start -D libid=3af012b1
```

Stopping a library

It is possible to stop a library manually by using the stop command of arkc. For instance:

```
arkc -library -stop -[D|I<input file>] <parameters...>
```

This command accepts the same parameters as the start command. For instance, the following commands are valid:

```
arkc -library -stop -D name=[HP SureStore 1/9 DLT]
arkc -library -stop -D libid=3af012b1
```

11. Miscellaneous commands

This chapter of the arkc manual contains commands that could not easily be grouped with others. These commands will be introduced in no special order.

Tape recycling: the recycle command

The recycle command of arkc allows the manual recycling of a tape. If applied to a pool, the recycle command will recycle all the tapes contained in the pool.

It is important to understand the distinction between the recycle command and the delete command.

Deleting a tape means removing its entry in the Arkeia database. The data contained on the tape is therefore unavailable to Arkeia and cannot be accessed, unless the tape data is re-imported into the Arkeia database, which is a long and time-consuming process. For more information on the `delete` command, please refer to the chapter named *The delete command of arkc* on page *33*, and especially to the section named Tape deletion.

Recycling a tape means the tape-related information is still contained in the Arkeia database, but the tape data itself can be erased, for instance for a newer backup. In effect, the tape is marked as *available* for all future backup operations.

The `recycle` command has the following syntax:

```
arkc -tape -recycle -[D|I<input file>] <parameters...>
```

The parameters that can be accepted by this command are:

- `name/tpid`**(Required)**
  The `name` and `tpid` parameters contain, respectively, the name and the ID of the tape to be recycled. Please note that only one parameter is needed. For instance, the following commands are equivalent, assuming `3b98c4b3` is the correct ID for the tape named `DLT004x`:

  ```
  arkc -tape -recycle -D name=DLT004x
  arkc -tape -recycle -D tpid=3b98c4b3
  ```

- `plname/plid`(Optional)
  The `plname` and `plid` parameters contain, respectively, the name and the ID of the tape pool to be recycled. Please note that only one parameter is needed. If a tape pool is recycled, all the tapes it contains are recycled and considered as available for future operations. For instance, the following commands would recycle the tape pool named `VXA`, whose ID is `3aaf90a4`:

  ```
  arkc -tape -recycle -D plname=VXA
  arkc -tape -recycle -D plid=3aaf90a4
  ```

Recycling duplication tapes with recycle_tpused

The Arkeia tape duplication object (`duplication`) provides a mechanism to notify if a tape has been modified since its last duplication. Internally, Arkeia stores a special file named `tpused.lst` which is used to track the tapes that have been duplicated.

By recycling (deleting) one entry in this file, it is possible to indicate that a tape has been modified since its last duplication and should probably be duplicated again.

The proper syntax for this recycling operation is the following one:

```
arkc -duplication -recycle_tpused -[D|I<input file>] <parameters...>
```

The only parameter required for this operation is the following one:

- name(**Required**)
  The name parameter should contain the name of the duplication object whose entry should be recycled. For instance:

  ```
  arkc -duplication -recycle_tpused -D name=CAD_Dup
  ```

  The parameter above indicates that the source tape of the CAD_Dup duplication object has been modified and should be duplicated again, if the data it contains is important enough to warrant a duplication.

  Please note that this parameter can be replaced by the dupid parameter, which should contain the ID of the same object. The value of dupid, like the value of name, is assigned to the duplication object during its creation. It is not necessary to set both parameters.

For more information on the tape duplication object, please refer to the section named *Tape duplication creation* on page *31* of this manual.

Writing a label on a tape with the write command

The write command writes a new Arkeia header on a tape. The correct syntax for this command is the following:

```
arkc -drive -write -[D|I<input file>] <parameters...>
```

The parameters that can be accepted by this command are the following ones:

- name/drvid(**Required**)
  The name and drvid parameters respectively contain the name or the ID of the drive containing the tape whose header should be written. It is possible to use or the other to designate a specific drive with the write command. For instance:

  ```
  arkc -drive -write -D name=[DLT 8000] -tpname=JackTape
  arkc -drive -write -D drvid=3aeffff3 -tpname=JackTape
  ```

  Please note that the two commands above are equivalent, assuming 3aeffff3 is the correct ID for the drive named DLT 8000. For more information on the tpname parameter, please see below.

- tpname(**Required**)
  The tpname parameter should contain the new header (name) of the tape. For an example of the syntax of this parameter, please see the name/drvid parameter examples above.

Please note: arkc does not allow the header of a tape to be written if the tape already has an existing header. To force the header to be written to the tape using the write command, use the overwrite parameter (see below).

- overwrite(Optional)
  Using the overwrite parameter, it is possible to delete the existing header of a tape, and replace it with the new header. For instance:

  ```
  arkc -drive -write -D name=[DLT 8000] -tpname=JackTape
  -overwrite
  ```

  Please note that the command above should be entered all on one line. This replaces the existing header of the tape introduced in the drive DLT 8000 by the header JackTape.

Checking the label on a tape with the read command

The read command checks if a tape is present in a given tape drive. If this is the case, it will then display the tape header.

The syntax for this command is the following:

```
arkc -drive -read -[D|I<input file>] <parameters...>
```

This command accepts only one parameter, drvname (or drvid), which is detailed below:

- drvname/drvid(**Required**)
  The parameter drvname contains the name of the drive that should be checked for the presence of a tape. For instance:

  ```
  arkc -drive -read -D drvname=ecrix_2
  ```

  The command above indicates that the drive to be checked for the presence of a tape is ecrix_2. If drvid should be used, the following command would be equivalent to the command above, assuming the ID of the drive ecrix_2 is 3aaf7e63:

  ```
  arkc -drive -read -D drvid=3aaf7e63
  ```

It is possible to filter the output of the read command, with the following filters:

- tpnameDisplays only the name of the tape present in the drive.
- tpidDisplays only the ID of the tape present in the drive.

For instance, the following command will display the ID of the tape present in a drive:

```
arkc -drive -read -D name=3RD_DLT -Ftpid
```

Modifying a job with the connect command

The connect command allows a job in process (backup or restoration) to be modified *on the fly* while it is being performed by Arkeia.

This command is very useful if the configuration of your Arkeia server is limited to one drive, as it allows the running task to proceed even if it is waiting for a new tape.

Modifying a running backup

The standard syntax to modify a running backup with the `connect` command is the following:

```
arkc -backup -connect -[D|I<input file>] <parameters...>
```

The following parameters can be used to modify a running backup job:

- `bksid`(**Required**)
  The `bksid` parameter contains the ID of the backup to be modified. This ID is returned by `arkc` when a backup is started with the `start` command.

- `wait`(Optional)
  The `wait` parameter is used to determine the behavior of `arkc` at the end of the backup operation. This parameter can accept two values: `YES` and `NO`. If the value `YES` is used, `arkc` will only accept new commands from the user after the backup is completed. If the value `NO` is used, `arkc` will immediately accept new commands from the user.

- `exec`(Optional)
  The `exec` parameter modifies the behavior of the running backup. The following operation can be entered as the value of this parameter:

  `ABORT`Stops the current backup.
  `ADDSK`Add a new Savepack to the current backup.
  `SETTAPE`Indicates that a new tape has been inserted.
  `SETMODE`Modifies the mode of the backup.

  For instance, the following command would abort the running backup with the ID `3a6718dc`:

  ```
  arkc -backup -connect -D bksid=3a6718dc exec=ABORT
  ```

- `drvname`(Optional)
  The parameter `drvname` determines the drive that contains the new tape. This parameter is **required** if the `exec` parameter has the value `SETTAPE`. Please refer to the `exec` parameter, above, for more information. For instance:

  ```
  arkc -backup -connect -D bksid=3a6718dc exec=SETTAPE
  drvname=ecrix_2
  ```

  The command above indicates to the running backup with an ID of `3a6718dc` that a new tape has been inserted in the drive named `ecrix_2` and can be used

to save the data being backed-up. Please note that this command should be placed all on one line.

- mode(Optional)
  The parameter mode should only be used if the exec parameter is set to SET-MODE. There are two possible values for mode, which are: STANDARD (the job will be stopped once the backup is finished) and CONTINUOUS (the job will stay active even after the end of the backup).

- skname(Optional)
  The parameter skname should only be used if the exec parameter is set to ADDSK (add a Savepack). The value of skname should then be the name of the Savepack to be added to the running job. For instance:

  ```
  arkc -backup -connect -D bksid=3a6718dc exec=ADDSK
  skname=JackFiles
  ```

  The command above adds the Savepack named JackFiles to the running backup designated by the ID: 3a6718dc. Please refer to the exec parameter above for more information.

Modifying a running restoration

The standard syntax to modify a running restoration with the connect command is the following:

```
arkc -restore -connect -[D|I<input file>] <parameters...>
```

The following parameters can be used to modify a running restoration:

- bksid(**Required**)
  The bksid parameter contains the ID of the restoration to be modified. This ID is returned by arkc when a restoration is started with the start command.

- wait(Optional)
  The wait parameter is used to determine the behavior of arkc at the end of the restoration. This parameter can accept two values: YES and NO. If the value YES is used, arkc will only accept new commands from the user after the restoration is completed. If the value NO is used, arkc will immediately accept new commands from the user.

- exec(Optional)
  The exec parameter modifies the behavior of the restoration. The following operation can be entered as the value of this parameter:

  ABORTStops the current restoration.

SETTAPEIndicates that a new tape has been inserted.
SETDRIVESet a drive for the current restoration.

For instance, the following command would abort the running restoration with the ID 3a6729dc:

```
arkc -restore -connect -D rstid=3a6729dc exec=ABORT
```

- drvname(Optional)
  The parameter drvname indicates either (a) the drive that contains the new tape (if exec has the value SETTAPE) or (b) the drive that should be used for the restoration (if exec has the value SETDRIVE). This parameter is **required** if the exec parameter has the value SETTAPE or SETDRIVE. Please refer to the exec parameter, above, for more information. For instance:

  ```
  arkc -restore -connect -D rstsid=3a6729dc exec=SETDRIVE
  drvname=ecrix_2
  ```

  The command above indicates to the restoration with an ID of 3a6729dc that the drive to be used for the operation is the ecrix_2 drive. Please note that this command should be placed all on one line.

Finding a file with the where command

Finding a file that should be restored can be a time-consuming process. The where command can help in this process. This command only applies to the file object.

The correct syntax for this command is the following:

```
arkc -file -where -[D|I<input file>] <parameters...>
```

The command where searches the internal Arkeia database and tries to find a match to the search criteria that were given.

This command accepts the following parameter:

- file(**Required**)
  The file parameter contains the name of the file (or directory) that should be searched for. The search criteria should be in the following format:

  ```
  machine:directory/directory/file
  ```

  For instance:

  ```
  arkc -file -where -D file=pluto:/home
  ```

  The command above will start a search in the Arkeia database for the directory /home that was backed-up from the machine named pluto.

The output of the where command can be modified with the following filters:

- tpnameDisplays only the name of the tape(s) where the files are found.

- tpidDisplays only the ID of the tape(s) where the files are found.

The rest of this manual will describe the arkc commands, divided among the different objects.

12. The tape object

This chapter of the arkc manual will detail all the possible commands that can be applied to the tape object. As such, it contains all the tape-related commands, such as create, delete, etc. that have been detailed in the previous chapters.

Tape creation

The creation of a tape with arkc, through the create command, uses the following syntax:

```
arkc -tape -create -[D|I] <parameters>
```

Please refer to the section named *arkc options: general* on page *10*, for more information on the different options that can be used with arkc.

The parameters that can be used with the create command are the followings:

- name(**Required**)
  The name parameter indicates the name which is going to be used by Arkeia to refer to the new tape. This parameter is required by the create command. The correct syntax for this parameter is: name=XXXXX, where XXXXX is the name (string) to assign to the new tape.
  For instance:

  ```
  name=DemoTape
  name=DAT_Database
  ```

- type(**Required**)
  The type parameter indicates the type of the tape being created. This parameter is required by the create command. Please note that the names should be entered precisely as shown below and that, whenever a name contain spaces, it should be enclosed with square brackets [...], after the type= function. For instance: type=[EXB 8500] is a valid syntax, while type=exb 8500 or type=Exb8500 are not. The available tape types are listed below:

```
AIT-1
AIT-1XL
AIT-2
DAT-DDS4
```

```
DAT-DDS3
DAT-120
DAT-60
DAT-90
D3-10GB
D3-25GB
D3-50GB
EXB 8500
EXB 8505XL
EXB MAMMOTH
EXB MAMMOTH 2
NULL
FILE 20MB
FILE 50MB
FILE 100MB
FILE 500MB
DLT 2000
DLT 2000XT
DLT 4000
DLT 7000
DLT 8000
DLT1
SDLT
MAGSTAR MP
MAGSTAR MP-C
MAGSTAR MP-CXL
DTF GW 730L
QIC 50GB
QIC 25GB
QIC 16GB
QIC 13GB
QIC 4GB
QIC 2GB
3590 CART
9840 CART
LTO Ultrium
VXA-V17
VXA-V6
```

Please note: It is possible to obtain this list of available tape types by entering the following command:

```
arkc -tape -type
```

For more information about the `type` command, please refer to the chapter named *The informational commands and objects of arkc* on page *48* of this manual.

- `comment`(Optional)
  The `comment` parameter can be used to insert a short string to offer more information to the user about the tape. The comment is displayed within the Arkeia GUI, or with the `-list` command of `arkc` (Please see the command `-list`, below, for more information). The correct syntax for this parameter is as follows:

  ```
  comment=[This tape is for database backup only]
  ```

Please note that a comment containing spaces should be placed between square brackets `[...]` as shown above.

- `firstnum` ... `lastnum`(Optional)
  `firstnum` and `lastnum` allow the user to define several identical tapes with only two commands. To do this, it's only necessary to enter the number of the first tape (after `firstnum=`) and the number of the last tape (after `lastnum=`). For instance, the following entries in a data file would create five (5) tapes:

  ```
  # Create 5 tapes _________
  firstnum=1
  lastnum=5
  ```

  Please note that the name of the tapes will be the value assigned to the command `name=`, followed by the tape number. The value of `firstnum` can be different than 1.

- `plname`/`plid`(Optional)
  The parameters `plname` and `plid` indicates to which tape pool the new tapes should be added. The parameter `plname` indicates the pool chosen by its name, and `plid` indicates the pool chosen by its internal Arkeia ID. For instance:

  ```
  plname=Test_Pool
  ```

  In the example above, the tape created with `-tape -create` will be added to the tape pool named `Test_Pool`.

- `recycle_in`(Optional)
  The parameter `recycle_in` determines in which tape pool the tape created should be placed when it is recycled. There are two possible pools defined by default: `SCRATCH_POOL` (default recycling pool) and `CURRENT_POOL`. For instance, the parameter:

  ```
  recycle_in=CURRENT_POOL
  ```

  will recycle the tape created in example above to `Test_Pool`, if the tape was assigned to this pool when it was created. By default, the tape is recycled in the pool it was assigned to when it was created.

- `recycle_mode`(Optional)
  The parameter `recycle_mode` determines the type of recycling operation that will be applied to the tape being created. This parameter can take two value: `FIFO` (first in/ first out) or `LIFO` (last in/first out). If the value chosen is `FIFO`, the oldest recycled tape will be used for a backup. If the value is `LIFO`, the newest recycled tape is used for the backup. For instance:

```
recycle_mode=FIFO
```

Please note that the values of this parameter should be entered all in uppercase letters. The default value is FIFO.

- access(Optional)
  The parameter access defines the access rights for the tape created. The possible access rights are the following: WRITE, READ, RECYCLE, DELETE, CLEAN. The different rights are defined within the same access parameters as follows:

  ```
  access=[WRITE|READ|RECYCLE]
  ```

  The parameter above authorize the WRITE, READ and RECYCLE operations on the tape that was created, but not any other. The default rights on all tapes are: READ, WRITE, RECYCLE, and DELETE.

- voltag(Optional)
  The parameter voltag is applied only to the tape of the type FILE (see above for tape types. The value of this parameter is the path which will be used to create the file itself. For instance:

  ```
  # Directory for the tape
  voltag=/home/jack/arkeia/files/arkeia_file
  ```

  The parameter above will create an Arkeia FILE tape in the directory: /home/jack/arkeia/files/arkeia_file. Please note that this parameter is required by all FILE tapes. You cannot put more than one tape in a given directory.

Tape deletion

To delete a date with arkc, use the following syntax:

```
arkc -tape -delete -D <parameters>
```

Please note: Deleting an Arkeia tape does not affect the physical tape itself, it means that the entry of the tape in the Arkeia database will be deleted. The tape data won't be accessible anymore, since the Arkeia database will not have a record of the tape.

The only way to access the tape data again is to import the tape information back in the Arkeia database with the arkrstb utility, an operation which can be complex and time-consuming if several tapes have been deleted. For more information on this operation, and the utilities provided to perform it, please refer to the Arkeia

The following parameters can be used to delete a tape:

- name(**Required**)
  The name parameter indicates which tape should be deleted by arkc. For instance, the following command will delete the tape named Test_Tape_01:

  ```
  arkc -tape -delete -D name=Test_Tape_01
  ```

  To delete several tapes, you can either:

  (**a**) Enter several names on the command line:

  ```
  arkc -tape -delete -D name=Test_Tape_01 name=Test_Tape_02
  ```

  (**b**) Delete all the tapes contained in a pool of tape (if all the tapes to be deleted are assigned to the same pool):

  ```
  arkc -tape -delete -D plid=3a78354e
  ```

  For more information on tape pool deletion, please see the parameter plid below.

  or (**c**) perform a piping operation, such as:

  ```
  arkc -tape -list | grep Test_ | arkc -tape -delete -
  ```

  In the command above, arkc first makes a list of tapes, which is piped to, and parsed by, the UNIX grep command. In the example above, all tapes with a name starting with Test_ will be returned. The result is then piped back into arkc for deletion. Please note the usage of the - input option in this example.

- tpid(Optional)
  The parameter tpid can replace the name parameter that we have detailed above. The tpid value is assigned to a tape when it is first created by Arkeia. This value can be displayed for a given tape by entering the command:

  ```
  arkc -tape -list -D name=name_of_tape
  ```

  Since the tpid parameter is equivalent to the name parameter, the two commands below are equivalent (provided that 3a7834d7 is the correct tpid value of the tape Test_Tape_01):

  ```
  arkc -tape -delete -D name=Test_tape_01
  arkc -tape -delete -D tpid=3a7834d7
  ```

- plid(Optional)
  The plid parameter is one way to delete several tapes in one operation, as long as all the tapes to be deleted are included in the same tape pool. The

plid of a given tape can be obtained by entering the following command:

```
arkc -tape -list -D name=name_of_the_tape
```

Once obtained, it is possible to delete all the tapes contained in a given pool tape by entering the command:

```
arkc -tape -delete -D plid=3a78354e
```

In the example above, all the tapes contained within the tape pool ID 3a78354e will be deleted with one command.

Please note: Caution is advised when using the plid parameter while deleting several tapes, as the delete command will delete all the tapes contained in the tape pool. For more information on tape pool deletion, please see below.

Tape modification

To modify a tape's characteristics, the syntax of arkc is the following:

```
arkc -tape -modify -[D|I<input file>] <parameters...>
```

To modify a tape, it is necessary to supply to arkc its name or its tpid. For more information on these parameters, please refer to *The create command of arkc* on page *15*. To obtain a list of the names of the available tapes, it is always possible to enter the following command:

```
arkc -tape -list
```

The following tape parameters can be changed using the modify command: plname, plid (Tape Pool name and ID), recycle_in, recycle_mode (Recycling functions), access (Access rights of the tape), comment. These parameters are described in more detail in the chapter *The create command of arkc* on page *15* of this manual.

Listing files present on a tape with the list command

The object file is used to obtain a list of files contained on a given tape. When using the command arkc -file -list, it is therefore necessary to supply the name of the tape to be listed with the tpname parameter. The standard syntax is therefore:

```
arkc -file -list -D tpname=name_of_tape
```

For instance:

```
arkc -file -list -D tpname=DLT_140
```

The command above will list all the files contained on the tape named DLT_140. For more information on the file object, please refer to the second part of this manual.

It is possible to filter the output of -file -list command with the following filter:

- fileDisplays only the files present on the tape.

Obtaining information on tapes with the type command

The type command is designed to provide information on the type of objects that can, for instance, be created by arkc. Its syntax is as simple as the one of the list command:

```
arkc -object -type
```

We have seen several examples of this command in the previous sections of this manual.

For tapes, the command arkc -tape -type will list all the tape standards that can be managed by Arkeia. A complete list can be found in this chapter, in section *Tape creation* on page *77* or in the equivalent section (*Tape creation* on page *16*) of the create command.

Displaying additional tape information with statistics

It is possible to obtain detailed statistics on a given tape by using the statistics command of arkc. This command is especially useful when the time comes to retire a tape at the end of its life and delete it from the Arkeia database.

The syntax for the statistics command is the following one:

```
arkc -tape -statistics -[D|I<input file>] <parameters...>
```

The possible parameters of the statistics command are the following:

- name/tpid(**Required**)
  The name and tpid parameters contain, respectively, the name and the ID of the tape whose statistics should be displayed. Please note that only one parameter is needed. For instance, the following commands are equivalent, assuming 3b98c4b3 is the correct ID for the tape named DLT00:

  ```
  arkc -tape -statistics -D name=DLT00
  arkc -tape -statistics -D tpid=3b98c4b3
  ```

- plname/plid(Optional)
  The plname and plid parameters contain, respectively, the name and the ID of the tape pool to be analyzed by the statistics command. Please note that only one parameter is needed. If a tape pool is used, all the tapes it contains are analyzed and their statistics displayed. For instance, the following commands would display the statistics of the tape pool named VXA, whose ID is 3aaf90a4:

  ```
  arkc -tape -statistics -D plname=VXA
  arkc -tape -statistics -D plid=3aaf90a4
  ```

Displaying log information with the journal object

The `arkc journal` object is used to display the log of the Arkeia server. The correct syntax to use for this is the following:

`arkc -journal -command -D <parameters...>`

The different commands that can be used with the `journal` object are the following:

- `-journal -all`Displays all log information.
- `-journal -jtape`Displays only tape-related information.
- `-journal -jdrive`Displays only drive-related information.
- `-journal -jbackup`Displays only backup-related information.
- `-journal -jrestore`Displays only restoration-related information.

The rest of this section will detail the `-jtape` command listed above. Since all the commands accept the same parameters, these will be detailed at the end of the section.

Displaying the tape-related information

The syntax to display only the tape-related information contained in the Arkeia log is the following:

`arkc -journal -jtape [-D] <parameters...>`

For more information about the parameters that can be accepted by this command, please refer to the section below.

Possible parameters for the journal object and commands

The parameters that can be accepted by the `journal` object are detailed below. Please note that these parameters apply to all the commands that were detailed above (`all`, `jtape`, `jdrive`, and `jrestore`), and that they are all optional.

- `month`(Optional)
  The parameter `month` limits the display of the `journal` object to a specific month. The value of month can go from `1` (January) to `12` (December). For instance:

  ```
  arkc -journal -jtape -D month=10
  ```

  The command above will display all the tape-related information and messaes that took place during the month of October of the current year. By default, the month displayed by the `journal` object is the current month.

- `level`(Optional)
  The parameter `level` specifies the quantity of information that should be displayed by a command. This parameter can accept three different values: `%I` (information messages), `%W` (warning messages) and `%E` (error messages). The default value for this parameter is `%I%W%E`, which displays all the log messages (information, warnings and errors). For instance:

```
arkc -journal -jtape -D level=%W%E
```

The command above will only display the tape-related warnings and errors that were reported by the Arkeia server during the restoration operations. On the other hand, the *information* messages won't be displayed. The `level` parameter therefore limits the information to the most interesting messages, which can be quite useful to track the source of a problem.

- `filter`(Optional)
  The `filter` parameter adds or removes specific data from the information returned by the Arkeia log. This parameter accepts the following values: `%D` (date), `%T` (time), `%L` (level, see above), `%I` (information code) and `%S` (message string). The default value for the `filter` parameter is `%D%T%L%I%S`, meaning that all information returned by the log is displayed. For instance:

  ```
  arkc -journal -jtape -D filter=%I%S
  ```

  The command above would limit the information returned by the Arkeia log to the information code, followed by the message string.

Tape recycling: the recycle command

The `recycle` command of `arkc` allows the manual recycling of a tape. If applied to a pool, the `recycle` command will recycle all the tapes contained in the pool.

It is important to understand the distinction between the `recycle` command and the `delete` command.

Deleting a tape means removing its entry in the Arkeia database. The data contained on the tape is therefore unavailable to Arkeia and cannot be accessed, unless the tape data is re-imported into the Arkeia database, which is a long and time-consuming process. For more information on the `delete` command, please refer to the chapter named *The delete command of arkc* on page *33*, and especially to the section named Tape deletion.

Recycling a tape means the tape-related information is still contained in the Arkeia database, but the tape data itself can be erased, for instance for a newer backup. In effect, the tape is marked as *available* for all future backup operations.

The `recycle` command has the following syntax:

`arkc -tape -recycle -[D|I<`*`input file`*`>] <`*`parameters`*`...>`
The parameters that can be accepted by this command are:

- `name/tpid`(**Required**)
  The `name` and `tpid` parameters contain, respectively, the name and the ID of the tape to be recycled. Please note that only one parameter is needed. For instance, the following commands are equivalent, assuming `3b98c4b3` is the correct ID for the tape named `DLT004x`:

```
arkc -tape -recycle -D name=DLT004x
arkc -tape -recycle -D tpid=3b98c4b3
```

- plname/plid(Optional)
  The plname and plid parameters contain, respectively, the name and the ID of the tape pool to be recycled. Please note that only one parameter is needed. If a tape pool is recycled, all the tapes it contains are recycled and considered as available for future operations. For instance, the following commands would recycle the tape pool named VXA, whose ID is 3aaf90a4:

  ```
  arkc -tape -recycle -D plname=VXA
  arkc -tape -recycle -D plid=3aaf90a4
  ```

Writing a label on a tape with the write command

The write command writes a new Arkeia header on a tape. The correct syntax for this command is the following:

```
arkc -drive -write -[D|I<input file>] <parameters...>
```

The parameters that can be accepted by this command are the following ones:

- name/drvid(**Required**)
  The name and drvid parameters respectively contain the name or the ID of the drive containing the tape whose header should be written. It is possible to use or the other to designate a specific drive with the write command. For instance:

  ```
  arkc -drive -write -D name=[DLT 8000] -tpname=JackTape
  arkc -drive -write -D drvid=3aeffff3 -tpname=JackTape
  ```

  Please note that the two commands above are equivalent, assuming 3aeffff3 is the correct ID for the drive named DLT 8000. For more information on the tpname parameter, please see below.

- tpname(**Required**)
  The tpname parameter should contain the new header (name) of the tape. For an example of the syntax of this parameter, please see the name/drvid parameter examples above.

Please note: arkc does not allow the header of a tape to be written if the tape already has an existing header. To force the header to be written to the tape using the write command, use the overwrite parameter (see below).

- overwrite(Optional)
  Using the overwrite parameter, it is possible to delete the existing header of a tape, and replace it with the new header. For instance:

  ```
  arkc -drive -write -D name=[DLT 8000] -tpname=JackTape
  -overwrite
  ```

  Please note that the command above should be entered all on one line. This replaces the existing header of the tape introduced in the drive DLT 8000 by the header JackTape.

Checking the label on a tape with the read command

The read command checks if a tape is present in a given tape drive. If this is the case, it will then display the tape header.

The syntax for this command is the following:

```
arkc -drive -read -[D|I<input file>] <parameters...>
```

This command accepts only one parameter, drvname (or drvid), which is detailed below:

- drvname/drvid(**Required**)
  The parameter drvname contains the name of the drive that should be checked for the presence of a tape. For instance:

  ```
  arkc -drive -read -D drvname=ecrix_2
  ```

  The command above indicates that the drive to be checked for the presence of a tape is ecrix_2. If drvid should be used, the following command would be equivalent to the command above, assuming the ID of the drive ecrix_2 is 3aaf7e63:

  ```
  arkc -drive -read -D drvid=3aaf7e63
  ```

It is possible to filter the output of the read command, with the following filters:

- tpnameDisplays only the name of the tape present in the drive.
- tpidDisplays only the ID of the tape present in the drive.

For instance, the following command will display the ID of the tape present in a drive:

```
arkc -drive -read -D name=3RD_DLT -Ftpid
```

13. The pool object

This chapter of the manual details all the commands that can be applied to the pool (tape pool) object of arkc.

Pool creation

The creation of a tape pool with `arkc`, through the `create` command, uses the following syntax:

```
arkc -pool -create -[D|I] <parameters>
```

Please refer to *arkc options: general* on page *10*, for more information on the different options that can be used with `arkc`. The parameters that can be accepted by the `create` command when it is used to create a new tape pool are the following:

- `name`(**Required**)
  The `name` parameter indicates the name which is going to be used by Arkeia to refer to the new tape pool. This parameter is required by the `create` command. The correct syntax for this parameter is: `name=XXXXX`, where `XXXXX` is the name (string) to assign to the new pool. For instance:

  ```
  name=Safe_Pool
  ```

  creates a tape pool named: `Safe_Pool`. Please note that it is possible to assign a `name` that contains space, if this name is entered between square brackets: `[...]`. For instance:

  ```
  name=[R&D tape pool]
  ```

  The parameter above will create a Tape Pool named: `R&D tape pool`.

- `owner`(Optional)
  The `owner` parameter indicates which Arkeia user is the owner of the tape pool. By default, the value of `owner` is set to the user who opened the session. It is possible to specify a different user than the user currently logged in to the session. For instance, if the command:

  ```
  $ arkc -user -list
  ```

  returns the following list of users:

  `name=jsmith`(John Smith)
  `name=tchang`(Tina Chang)
  `name=root`

  The `owner` value can be either `jsmith`, `tchang` or `root`.

- `comment`(Optional)
  The `comment` parameter can be used to insert a short string to offer more information to the user about the tape pool being created. The comment is displayed in the Arkeia GUI, or with the `-list` command of `arkc` (For more information on this command, please refer to *Listing objects with the list command* on page *49*). The correct syntax for this parameter is as follows:

```
comment=[This group of tape should be placed in the safe]
```

Please note that comment containing spaces should be placed between square brackets `[...]` as shown above.

Tape pool deletion

Deleting a tape pool is almost the same as deleting a single tape.

The correct syntax to delete a tape pool with `arkc` is the following one:

```
arkc -pool -tape -D [name|plid]=<value>
```

The parameters that can be accepted while deleting a tape pool are the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the pool to be deleted. For instance:

  ```
  arkc -pool -delete -D name=Test_Pool
  ```

  A list of all the tape pools available on a given Arkeia server can be obtained with the command:

  ```
  arkc -pool -list
  ```

- `plid`(Optional)
  The `plid` parameter is the ID assigned to the tape pool when it was created by Arkeia. This ID value can be used in place of the `name` parameter described above. For instance, the two commands below are equivalent, assuming that `3a78354e` is the correct ID of `Test_Pool`:

  ```
  arkc -pool -delete -D name=Test_Pool
  arkc -pool -delete -D plid=3a78354e
  ```

Please note: To delete a tape pool, it is necessary to first delete all the tapes that it contains. A tape pool which is not *empty* of tapes cannot be deleted by `arkc`. Any attempt to delete a non-empty tape pool will produce an error message.

Displaying information on tape pools

It is possible to obtain detailed information on a specific tape pool by using the list command of arkc. For more information on this command, please refer to the chapter of this manual titled *The informational commands and objects of arkc* on page *48*, and especially to the section named: *Listing objects with the list command* on page *49*.

For instance, to list all the pools that have been defined on a server, it is possible to enter the command:

```
$ arkc -pool -list

name=scratch pool
name=Sony AIT2
name=MiniLTO
name=null
name=File
name=VXA
name=DAT-90
name=DAT-120
name=lto
```

By entering the name of the pool, `arkc` displays the relevant pool-related information:

```
$ arkc -pool -list -D name=VXA

NB_THREADS=000
PLID=3aaf90a4
NAME=VXA
<ITEM>
THREAD=000
NB_TAPES=00009
</ITEM>
```

In the example above, all the information on the tape pool named `VXA` is displayed, including its ID (line `PLID`), its name (`NAME`) and the number of tapes it contains `NB_TAPES`).

14. The savepack object

This chapter of the manual will detail all the `arkc` commands that can be applied to the `savepack` object.

Savepack creation

Within Arkeia, a *Savepack* is a group of files, directories, disks or even whole set of machines, which is used to simplify a backup operation. Instead of choosing individual files, an administrator can create a Savepack, name it, and use the Savepack to quickly and effortlessly create and launch repetitive backup operations.

To create a Savepack with `arkc` and its `create` command, use the following syntax:

```
arkc -savepack -create -[D|I|-] <parameters>
```

Please note: using `arkc -create -savepack` only creates an empty Savepack. To add files and trees to a new Savepack, it is necessary to use the command `arkc -savepack -modify` (See below).

The parameters that can be used while creating a Savepack are the following ones:

- `name`(**Required**)
  The `name` parameter assign a name to the Savepack that will be created by arkc. This parameter is required by the `create` command. The correct syntax for this parameter is: `name=XXXXX`, where `XXXXX` is the Savepack's name. For instance:
  `name=Accounting`
  creates a Savepack named: `Accounting`. If you'd like to insert spaces in the name of the Savepack, you should enter the name between square brackets `[...]`. For instance:
  `name=[CAD/CAM Savepack]`

- `comment`(Optional)
  This parameter is used to assign a descriptive comment to the Savepack being created. This comment can be useful to display additional information. The correct syntax for this parameter is:
  `comment=[Savepack for CAD machines]`

- `allowed_fs`(Optional)
  The parameter `allowed_fs` defines how Arkeia is going to process different file systems while backing up the Savepack. This parameter can accept the three following values:

  `NORMAL`
  `ALL`
  `ALL_EXCEPT_NFS`

  When `allowed_fs` is set to `NORMAL`, the back-up of a Savepack will only process normal file systems. The value `ALL` indicates that all file systems (local, NFS and others) should be backed-up. The value `ALL_EXCEPT_NFS` indicates that all file systems, except NFS, should be backed-up while Arkeia is backing-up a given Savepack.

- `retry`(Optional)
  This parameter defines the number of times Arkeia should try to connect to a remote machine, that contains a tree included in the Savepack being backed-up. By default, the value of `retry` is `3`. The possible values are numbers from `1` through `10`.

- `compress`(Optional)
  This parameter defines what type of compression should be applied to the files contained in the Savepack. The possible values for `compress` are the following:

  `LZ1_LZ3`

```
LZ1
LZ3
NO_COMPRESS
```

The default value for the `compress` parameter is `LZ1_LZ3`, which lets Arkeia choose the optimal compression scheme between `LZ1` and `LZ3`, based on the data contained in the file being backed-up. The value `NO_COMPRESS` turns the file compression function off.

- `crypt`(Optional)
  This parameter defines the type of encryption that will be applied to the files contained in a Savepack. The possible values for `crypt` are the following:

  ```
  DES_BLOWFISH
  DES
  BLOWFISH
  NO_CRYPT
  ```

  By default, the value of `crypt` is `DES_BLOWFISH`. Using the value `NO_CRYPT` turns off the encryption features of Arkeia. The encryption functions that are available within Arkeia are described in much more details in the Arkeia User Manual, which is available on Knox Software web site (http://www.arkeia.com).

- `symb_link`(Optional)
  The parameter `symb_link` determines the behavior of Arkeia when it encounters a UNIX symbolic link within the files contained in a Savepack. The possible values for this parameter are `YES` or `NO`. A value of `YES` indicates that Arkeia should follow the symbolic links and a value of `NO` indicates that symbolic links should not be followed. The default value for this parameter is `NO`.

Please note: The `symb_link` parameter should be used with caution, as symbolic links can sometimes reference each other in a loop. For instance, link `A` can point to link `B`, which, in turn, points back to link `A`. Such a situation creates an infinite loop, which will block Arkeia's normal operations. Caution is therefore strongly advised.

- `follow_fs`(Optional)
  The parameter `follow_fs` determines the behavior of Arkeia when it encounters a mounted file system contained within a Savepack to be backed-up. This parameter can accept two values: `YES` (back-up mounted file systems) and `NO` (do not back-up mounted file systems). By default, the value of `follow_fs` is `YES`.

- `inc_filter`(Optional)
  The `inc_filter` defines an inclusion filter that will be applied to the backup of a Savepack. Once a filter is defined, Arkeia will backup the Savepack files only if they match the filter. All other files will be ignored. The filter itself is a normal UNIX regular expressions. For instance:

  ```
  inc_filter=.*W..\.dat
  ```

  The filter above will only backup the files whose name contain an upper-case `W`, followed by any two characters and by the letters `.dat`. For instance, a file named `CAD_Project.W55.dat` would be backed-up with such a filter.

- `exc_filter`(Optional)
  The `exc_filter` defines an exclusion filter that will be applied to the backup of a Savepack. Once a filter is defined, Arkeia will not backup the Savepack files if they match the filter. All other files will be saved. The filter itself is a normal UNIX regular expressions. For instance:

  ```
  exc_filter=.*D..\.dat
  ```

  The filter above won't backup the files whose name contain an upper-case `W`, followed by any two characters and by the letters `.dat`. For instance, a file named `DataBase.D56.dat` would not be backed-up with such a filter in place.

- `cmd_before`(Optional)
  The parameter `cmd_before` defines a command that has to be executed before the Savepack is processed by Arkeia. Please note that the command will be executed on the client machine. This parameter accepts any command that can be executed, including the path if necessary. For instance:

  ```
  cmd_before=[rm -fv /home/jack/temp/*.tmp]
  ```

  This command will remove all files ending with `.tmp` in the directory `/home/jack/temp/` prior to the backup of the directory `/home/jack` contained in the Savepack.

- `do_cmd_before`(Optional)
  The parameter `do_cmd_before` defines the behavior of Arkeia if the command defined by the parameter `cmd_before` fails. This parameter can accept two values: `YES` or `NO`. If `do_cmd_before` is set to `YES`, the back-up will proceed even if the command failed. If it is set to `NO`, the backup of the Savepack will be aborted if the command launched before the backup fails.

- `cmd_after`(Optional)
  The parameter `cmd_after` defines a command that has to be executed after the

Savepack is processed by Arkeia. Please note that the command will be executed on the client machine. This parameter accepts any command that can be executed, including the path if necessary. For instance:

```
cmd_after=[dvrt -xc]
```

This means that the command `dvrt` will be executed once all the files contained in the Savepack have been backed-up.

- `do_cmd_after`(Optional)
  The parameter `do_cmd_after` defines the behavior of Arkeia if the file backup fails. This parameter can accept two values: `YES` or `NO`. If `do_cmd_after` is set to `YES`, the command defined by the parameter `cmd_after` will always be executed, even if the backup itself fails. If it is set to `NO`, the command defined by `cmd_after` will not be executed if the backup of the Savepack has failed.

Savepack deletion

The correct syntax to delete a Savepack is the following one:

```
arkc -savepack -delete -[D|I] <parameters...>
```

Where the parameters can be the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the Savepack to be deleted. For instance:

  ```
  arkc -savepack -delete -D name=WebDesign
  ```

  A list of all the savepacks available on a given Arkeia server can be obtained with the command: `arkc -savepack -list`

- `skid`(Optional)
  The `skid` parameter is the ID assigned to the Savepack when it was created by Arkeia. This ID value can be used instead of the `name` parameter described above. For instance, the two commands below are equivalent, assuming that `3a64332e` is the correct ID of the Savepack named `WebDesign`:

  ```
  arkc -savepack -delete -D name=WebDesign
  arkc -savepack -delete -D plid=3a64332e
  ```

Savepack modification

To modify a Savepack with arkc, the following syntax should be used:

```
arkc -savepack -modify -[D|I<input file>] <parameters...>
```

Please note that, when modifying a Savepack with `arkc`, it is necessary to supply the parameter `name` or the parameter `skid` (name or ID of the Savepack to be modified).

The following Savepack parameters can be changed through the `modify` command:

- `allowed_fs`(file systems allowed)
- `retry`(number of allowed retries)
- `compress`(compression scheme to be used for back-up)
- `crypt`(cryptographic function to be used)
- `symb_link`(follow-through of symbolic links)
- `follow_fs`(follow-through of file systems)
- `inc_filter`(inclusion filter)
- `exc_filter`(exclusion filters)
- `cmd_before`, `do_cmd_before` (pre-backup command)
- `cmd_after` and `do_cmd_after` (post-backup commands behavior)
- `comment`

For more information on these parameters, please refer to *The create command of arkc* on page *15*.

The main use of the `modify` command, when it applies to Savepack, is to add or remove trees, files and their relevant parameters. The rest of this section will therefore detail the modification of the Arkeia Savepacks through the use of the `modify` command of `arkc`.

The information regarding a specific tree have to entered in a configuration file, and each item should be separated by a pair of `<ITEM>...</ITEM>` statements. For instance, the following lines are valid tree information:

```
<ITEM>
tree_name=station01:/home/jack/mail
</ITEM>

<ITEM>
tree_name=station01:/home/jack/CADfiles
</ITEM>

<ITEM>
tree_name=station01:/home/jack/project/documentation
</ITEM>
```

These lines, inserted in a configuration file, would indicate that, on the computer named `station01`, the following directories and their contents should be backed up by Arkeia: `/home/jack/mail`, `/home/jack/CADfiles` and `/home/jack/project/documentation`.

The following parameters can be added to the `<ITEM>...</ITEM>` statements:

- tree_name(Optional)
  The tree_name parameter defines the new tree that will be added to a Savepack. This parameter has the following format:

  ```
  name_of_machine:/directory1/directory2/...
  ```

  For instance:

  ```
  tree_name=station01:/home/jack/project/programming/C_files
  ```

  Please note that, to delete a specific tree from within a Savepack, you need to use the parameter dtree_name on the command line. The parameter tree_name and its output FULLNAME (see example above) are equivalent (FULLNAME is the internal information contained within the Arkeia database).

  The examples above are valid for UNIX machines, running operating systems such as Linux or Solaris. For Windows machines, the complete syntax of tree_name is the following:

  ```
  tree_name=machine_name:drive_letter:/directory1/...
  ```

  For instance:

  ```
  tree_name=dune:c:/common/documents/sales/2000
  ```

  Please note the second : after the drive letter. Please note also that arkc requires normal slashes / instead of the standard inverted slashes \ of Windows

Please note: Never use the internal Arkeia name (FULLNAME) whenever you enter a command or edit an arkc input file. Please note as well that the name of a Savepack and the name of a tree are two different parameters. To add a new tree to an existing Savepack, it is therefore necessary to enter either, on the command line:

```
arkc -savepack -modify -D name=station01 tree_name=station01:/home/linda
```

or, within an input file:

```
# Saves Linda home directory on computer station01

<ITEM>
tree_name=station01:/home/linda
</ITEM>
```

Any other syntax than the two shown above will be rejected by arkc. Please note that the two possibilities above are equivalent.

- tree_family(Optional)
  Arkeia uses a sequential procedure to save the trees contained within a Savepack. In order to benefit from Arkeia's advanced parallelism functions, it is necessary to group the different trees in different *families*, using the tree_family parameter. This allows Arkeia to save these tree families at the same time. The values that can be accepted by tree_family are numerical, from 0 to 128. For instance:

```
# Savepack configuration file for station01
# This contains Jack Smith files

<ITEM>
tree_name=station01:/home/jack/mail
tree_family=1
</ITEM>

<ITEM>
tree_name=station01:/home/jack/CADfiles
tree_family=0
</ITEM>

<ITEM>
tree_name=station01:/home/jack/project/documentation
tree_family=0
</ITEM>
```

  The example above defines two tree families: 0 and 1. Between these two families, it is now possible to define the importance of each, and the order in which they are going to be saved. Please see below, the tree_priority and tree_chain parameters for more information.

- tree_priority(Optional)
  The parameter tree_priority defines the priority to be assigned to a particular tree. This parameter accepts numerical values, starting from 0 (highest priority) to 100 (lowest priority). Trees are then backed up according to their priorities. For instance, starting with the example above:

```
# Savepack configuration file for station01
# This contains Jack Smith files

<ITEM>
tree_name=station01:/home/jack/mail
tree_family=1
tree_priority=1
</ITEM>

<ITEM>
tree_name=station01:/home/jack/CADfiles
tree_family=0
tree_priority=0
</ITEM>

<ITEM>
tree_name=station01:/home/jack/project/documentation
```

```
tree_family=0
tree_priority=0
</ITEM>
```

In the example above, the two families `0` and `1` are given different levels of priority. Tree family number `0` (`/home/jack/CADfiles` and `/home/jack/project/documentation`) are given a higher priority to reflect their higher importance, while tree `1` is given a lower priority. This gives the result that both families will be backed up in parallel by Arkeia, while giving a higher priority to the `documentation` and `CADfiles` directories. The default value of `tree_priority` is `50`.

- `tree_chain`(Optional)
  The parameter `tree_chain` is used to create a dependency between trees contained in a Savepack. If two trees are created with the same `tree_chain` value (from `1` to `128`), a dependency is created between the two trees. To control the backup order, the `tree_priority` should be used (please see above for more information). Please note that a value of `0` means there are no chaining (no dependencies) between trees.

- `tree_type`(Optional)
  The parameter `tree_type` defines the type of tree that will be a part of the Savepack. This parameter can accept three different values, which are:

  `TREE`(standard object file or directory)
  `OBJECT`(the result of a command should be saved)
  `RAW`(physical device, such as `/dev/fd0`)

  This parameter determines which other parameters can be applied to the tree. Please see below for more examples. By default, the value of `tree_type` is `TREE`.

The following parameters can be used only if the `tree_type` parameter is set to `TREE`. They are the equivalent, for a single tree, of the general commands available for a compete Savepack. As such, they will not be detailed: please refer to *The create command of arkc* on page *15* for more information on these different parameters.

- `tree_comment`(Optional)
  The parameter `tree_comment` assigns a descriptive information to the tree. For more information on this parameter, please refer to *Savepack creation* on page *22*.

- `tree_retry`(Optional)
  The parameter `tree_retry` assigns a retry value to the tree. For more information, please refer to the `retry` parameter in *Savepack creation* on page *22*.

- `tree_compress`(Optional)
  The parameter `tree_compress` chooses a compression scheme for the tree. For more information, please refer to the `compress` parameter in *Savepack creation* on page *22*.

- `tree_crypt`(Optional)
  The parameter `tree_crypt` chooses a cryptographic scheme for the tree. For more information, please refer to the `crypt` parameter in *Savepack creation* on page *22*.

- `tree_cmd_before`(Optional)
  The parameter `tree_cmd_before` describes a command that should be executed before the tree is backed-up. For more information, please refer to the `cmd_before` parameter in *Savepack creation* on page *22*.

- `tree_do_cmd_before`(Optional)
  The parameter `tree_do_cmd_before` describes the behavior of Arkeia in the case of a failure of the command described in `tree_cmd_before`. For more information, please refer to the `do_cmd_before` parameter in *Savepack creation* on page *22*.

- `tree_cmd_after`(Optional)
  The parameter `tree_cmd_after` assigns a command to be executed after the back-up of the tree by Arkeia. For more information, please refer to the `cmd_after` parameter in *Savepack creation* on page *22*.

- `tree_do_cmd_after`(Optional)
  The parameter `tree_do_cmd_after` describes the behavior of the command entered in `tree_cmd_after` in the case of a failure of the tree backup. For more information, please refer to the `do_cmd_after` parameter in *Savepack creation* on page *22*.

- `tree_allowed_fs`(Optional)
  The parameter `tree_allowed_fs` determines the behavior of the Arkeia backup in case the tree contains NFS links. For more information, please refer to the `allowed_fs` parameter in *Savepack creation* on page *22*.

- `tree_follow_fs`(Optional)
  The parameter `tree_follow_fs` determines if Arkeia should follow shared file systems (for instance: NFS) while saving the tree. For more information, please refer to the `follow_fs` parameter in *Savepack creation* on page *22*.

- `tree_inc_filter`(Optional)
  The parameter `tree_inc_filter` determines an include filter for the tree backup. For more information, please refer to the `inc_filter` parameter in *Savepack creation* on page *22*.

- `tree_exc_filter`(Optional)
  The parameter `tree_inc_filter` determines an exclusion filter for the tree backup. For more information, please refer to the `exc_filter` parameter on page 24.

Listing avalaible savepacks wit the list command

For more information on the list command, please refer to the chapter named *The informational commands and objects of arkc* on page *48*, and especially the section: *Listing objects with the list command* on page *49*.

15. The drive object

This chapter will detail the `arkc` commands applicable to the `drive` object.

Drive creation

The creation of a tape drive with `arkc`, through the `create` command, uses the following syntax:

```
arkc -drive -create -[D|I] <parameters>
```

The parameters that can be used while creating a tape drive are the following ones:

- `name`(**Required**)
  The `name` parameter indicates the name which is going to be used by Arkeia to refer to the new tape drive. This parameter is required by the `create` command. The correct syntax for this parameter is: `name=XXXXX`, where `XXXXX` is the name (string) to assign to the new drive. For instance:

  ```
  name=Accounting
  ```

  The parameter above creates a drive named: `Accounting`.

- type(**Required**)
  The type parameter indicates which type of tape drive is going to be created by arkc. Please note that this parameter is required by the create command. Here is a list of available types of tape drive that are valid for this parameter:

  ```
  NULL
  FILE
  STD_AIT
  STD_DAT
  STD_EXABYTE
  STD_DLT
  STD_DTF
  DAT_SONY_7000
  AIT_AUTOLOADER
  STD_QIC
  STD_CARTRIDGE
  REDWOOD_SD3
  LTO
  MAGSTAR_MP
  ECRIX_VXA
  9840
  ```

  For instance, to create a DLT tape drive, a correct command syntax would be the following one:

  ```
  arkc -drive -create -D name=DLT02 type=STD_DLT
  rewind_dev=/dev/st0
  ```

  Please note that this command should be entered all on one line. You can also refer to the parameter rewind_dev below).

- rewind_dev(**Required**)
  The parameter rewind_dev is used to define the system device used by the operating system to access the tape drive. This device is usually used by the mt UNIX command. For instance, under most Linux systems, this device is /dev/st0.

- access(Optional)
  This parameter is used to define the access rights to be assigned to the device being created. These rights can be READ, WRITE, RECYCLE and DELETE. To authorize certain access rights, these should be included as values of the access parameter, for instance:

  ```
  access=[READ|WRITE]
  ```

  The parameter above indicates that the drive being created can be accessed for read/write operations, but not for deleting or recycling. By default, all four type of rights are granted on a new tape drive.

- comment(Optional)
  This parameter is used to assign a descriptive comment to the tape drive being created. This comment can be useful to display additional information related to the tape drive, as, for instance, its location, purpose, etc. The correct syntax for this parameter is:
  comment=[DLT tape drive for CAD machines]

Drive deletion

Deleting a drive allows the system administrator to quickly reconfigure Arkeia, in the case maintenance or replacement is needed for a hardware unit.

The correct syntax to delete a drive is the following one:

```
arkc -drive -delete -[D|I] <parameters...>
```

Where the parameters can be the following ones:

- name(**Required**)
  The name parameter should contain the name of the drive to be deleted. For instance:

  ```
  arkc -drive -delete -D name=DLT02
  ```

  A list of all the tape drives available on a given Arkeia server can be obtained with the command:

  ```
  arkc -drive -list
  ```

- drvid(Optional)
  The drvid parameter is the ID assigned to the tape drive when it was created by Arkeia. This ID value can be used instead of the name parameter described above. For instance, the two commands below are equivalent, assuming that 3bc419cd is the correct ID of the tape drive named DLT02:

  ```
  arkc -pool -delete -D name=DLT02
  arkc -pool -delete -D plid=3bc419cd
  ```

Drive modification

To modify a tape drive characteristics, the syntax of arkc is the following:

```
arkc -drive -modify -[D|I<input file>] <parameters...>
```

To modify a tape drive, it is necessary to supply to arkc its name or its drvid. For more information on this parameter, please refer to *The create command of arkc* on page *15*. To obtain a list of the names of the available tape drives, it is always possible to enter the following command:

```
arkc -drive -list
```

To obtain a detailed list of the existing characteristics of a specific tape drive (including the drvid parameter), it is possible to enter the command:

```
arkc -drive -list -D name=name_of_drive
```

For instance, the following command will return all relevant information on the tape drive named Sony AIT2:

```
arkc -drive -list -D name=[Sony AIT2]

PID=19533
DRVID=3bc1d31b
COMMENT=Sony AIT-2 drive plugged on Zeus
NODE=zeus
OWNER=jsmith
DRV_NUM=0
LIBID=0
DRV_DENY=0
STATUS=E
TIME_BEFORE_CLEAN=32767
USAGE_TIME=14924
NB_LOADS=0
CONTROL_DEV=/dev/st0
NONREWIND_DEV=/dev/null
REWIND_DEV=/dev/st0
DRV_TYPE=STD_AIT
NAME=Sony AIT2
TIME_AFTER_CLEAN=14924
```

The following tape drive parameters can be changed using the modify command: rewind_dev (tape drive UNIX controller), access (access rights of the tape drive), and comment. These parameters are described in more detail in the section above: *Drive creation* on page *100*.

Obtaining information on drives with the type command

The type command is designed to provide information on the type of objects that can, for instance, be created by arkc. Its syntax is as simple as the one of the list command:

```
arkc -object -type
```

We have seen several examples of this command in the previous chapters and sections of this manual.

For drives, the command arkc -drive -type will list all the drive standards that can be managed by Arkeia.

Displaying drive-related logs with the journal object

The arkc journal object is used to display the log of the Arkeia server. The correct syntax to use for this is the following:

```
arkc -journal -command -D <parameters...>
```

The different commands that can be used with the journal object are the following:

- -journal -allDisplays all log information.
- -journal -jtapeDisplays only tape-related information.

- `-journal -jdrive`Displays only drive-related information.
- `-journal -jbackup`Displays only backup-related information.
- `-journal -jrestore`Displays only restoration-related information.

The rest of this section will detail the `-jdrive` command listed above.

Displaying the drive-related information

The syntax to display only drive-related information from the Arkeia log is the following:

```
arkc -journal -jdrive [-D] <parameters...>
```

Possible parameters for the journal object and -jdrive command

The parameters that can be accepted by the `journal` object are detailed below. Please note that these parameters apply to all the commands that were listed above (`all`, `jtape`, `jdrive`, and `jrestore`), and that they are all optional.

- `month`(Optional)
  The parameter `month` limits the display of the `journal` object to a specific month. The value of month can go from `1` (January) to `12` (December). For instance:

  ```
  arkc -journal -jdrive -D month=10
  ```

  The command above will display all the drive-related information that was logged during the month of October of the current year. By default, the month displayed by the `journal` object is the current month.

- `level`(Optional)
  The parameter `level` specifies the quantity of information that should be displayed by a command. This parameter can accept three different values: `%I` (information messages), `%W` (warning messages) and `%E` (error messages). The default value for this parameter is `%I%W%E`, which displays all the log messages (information, warnings and errors). For instance:

  ```
  arkc -journal -jdrive -D level=%W%E
  ```

  The command above will only display the drive-related warnings and errors that were reported by the Arkeia server. On the other hand, the information messages won't be displayed. The `level` parameter therefore limits the information to the most interesting messages, which can be quite useful to track the source of a problem.

- `filter`(Optional)
  The `filter` parameter adds or removes specific data from the information returned by the Arkeia log. This parameter accepts the following values: `%D` (date), `%T` (time), `%L` (level, see above), `%I` (information code) and `%S` (message string). The default value for the `filter` parameter is `%D%T%L%I%S`, meaning

that all information returned by the log is displayed. For instance:

```
arkc -journal -jdrive -D filter=%I%S
```

The command above would limit the drive information returned by the Arkeia log to the information code, followed by the message string.

Checking the label on a tape with the read command

The `read` command checks if a tape is present in a given tape drive. If this is the case, it will then display the tape header.

The syntax for this command is the following:

`arkc -drive -read -[D|I<`*`input file`*`>] <`*`parameters`*`...>`
This command accepts only one parameter, `drvname` (or `drvid`), which is detailed below:

- `drvname/drvid`(**Required**)
  The parameter `drvname` contains the name of the drive that should be checked for the presence of a tape. For instance:

  ```
  arkc -drive -read -D drvname=ecrix_2
  ```

  The command above indicates that the drive to be checked for the presence of a tape is `ecrix_2`. If `drvid` should be used, the following command would be equivalent to the command above, assuming the ID of the drive `ecrix_2` is `3aaf7e63`:

  ```
  arkc -drive -read -D drvid=3aaf7e63
  ```

It is possible to filter the output of the `read` command, with the following filters:

- `tpname`Displays only the name of the tape present in the drive.
- `tpid`Displays only the ID of the tape present in the drive.

For instance, the following command will display the ID of the tape present in a drive:

```
arkc -drive -read -D name=3RD_DLT -Ftpid
```

Writing a label on a tape with the write command

The `write` command writes a new Arkeia header on a tape. The correct syntax for this command is the following:

`arkc -drive -write -[D|I<`*`input file`*`>] <`*`parameters`*`...>`
The parameters that can be accepted by this command are the following ones:

- `name/drvid`(**Required**)
  The `name` and `drvid` parameters respectively contain the name or the ID of the

drive containing the tape whose header should be written. It is possible to use or the other to designate a specific drive with the `write` command. For instance:

```
arkc -drive -write -D name=[DLT 8000] -tpname=JackTape
arkc -drive -write -D drvid=3aeffff3 -tpname=JackTape
```

Please note that the two commands above are equivalent, assuming `3aeffff3` is the correct ID for the drive named `DLT 8000`. For more information on the `tpname` parameter, please see below.

- `tpname`(**Required**)
  The `tpname` parameter should contain the new header (name) of the tape. For an example of the syntax of this parameter, please see the `name/drvid` parameter examples above.

Please note: `arkc` does not allow the header of a tape to be written if the tape already has an existing header. To force the header to be written to the tape using the `write` command, use the `overwrite` parameter (see below).

- `overwrite`(Optional)
  Using the `overwrite` parameter, it is possible to delete the existing header of a tape, and replace it with the new header. For instance:

```
arkc -drive -write -D name=[DLT 8000] -tpname=JackTape
-overwrite
```

Please note that the command above should be entered all on one line. This replaces the existing header of the tape introduced in the drive `DLT 8000` by the header `JackTape`.

16. The drivepack object

This chapter will detail all the `arkc` commands that can be applied to the `drivepack` object.

Drivepack creation

A Drivepack is a group of drives that is created within Arkeia in order to ease the management of large backups that may span several tapes/drives. Drivepacks allow the user to group drives and use them, as a single entity, for all backup operations.

Drivepacks are created with the following command:

```
arkc -create -drivepack -[D|I|-] <parameters>
```
The parameters that can be accepted while creating a Drivepack are the following:

- `name`(**Required**)
  The `name` parameter assign a name to the Drivepack that will be created by arkc. This parameter is required by the `create` command. The correct syntax for this parameter is: `name=XXXXX`, where `XXXXX` is the Drivepack's name. For instance:

  ```
  name=3rd_Floor
  ```

  creates a Drivepack named: `3rd_Floor`. If you'd like to insert spaces in the name of the Drivepack, you should enter the name between square brackets `[...]`. For instance:
  ```
  name=[DLT Drivepack]
  ```

- `comment`(Optional)
  This parameter is used to assign a descriptive comment to the Drivepack being created. This comment can be useful to display additional information. The correct syntax for this parameter is:

  ```
  comment=[DAT Drivepack for small backups]
  ```

- `nbdrive`(Optional)
  This parameter defines the number of drives that can be used by the Drivepack for a given operation. By default, the value of `nbdrive` is the number of drives that are contained in the Drivepack. If `nbdrive` has a lower value, this means a group of drives within the Drivepack will be used for the first task, while some drives will be reserved for other tasks. For instance, if a Drive pack contains 5 drives, it is possible to reserve three of these (`nbdrive=3`) for backup and keep 2 drives for restoration operations. Combining this parameter with the `priority` parameter (see below for more information) offers a great flexibility to system administrators in deciding which drive will be used for what purpose.

  Please refer to the information below (`drvname` and `drvid` parameters) for more details on adding new drives to a Drivepack.

- `drvname`(Optional)
  To add drives to a Drivepack, it is possible to use either the `drvname` or the `drvid` parameters. Both these parameters should be contained in an input file, with each drive to be added to the Drivepack contained in a pair of `<ITEM>...</ITEM>` statements. For instance, the following lines of an input file would add the drive named `DLT01` in a Drivepack:

  ```
  # Comment: First DLT drive of drivepack.
  <ITEM>
  ```

```
drvname=DLT01
priority=1
</ITEM>
```

Adding several drives to a Drivepack is therefore a question of adding several <ITEM>...</ITEM> statements into the same input file, each statement containing different drvname and priority parameters.

- drvid(Optional)
  To add drives to a Drivepack, it is possible to use either the drvname or the drvid parameters. Both these parameters should be contained in an input file, with each drive to be added to the Drivepack contained in a pair of <ITEM>...</ITEM> statements. For instance, the following lines of an input file would add the drive with the ID of DLT01 in a Drivepack:

  ```
  # Comment: First DLT drive of drivepack.
  <ITEM>
  drvid=3bc419c3
  priority=1
  </ITEM>
  ```

  Adding several drives to a Drivepack is therefore a question of adding several <ITEM>...</ITEM> statements into the same input file, each statement containing different drvname and priority parameters.

Please note: It is possible to add drives to a Drivepack by using the -D option, followed by the drvname or the drvid parameters. But using the command line does not allow the use of a specific priority parameter for each drive. In most cases, it is highly recommended to use an input file containing the drive parameters, with the -I command-line option, in order to set priority for each specific drive.

For example, the following line will create a new Drivepack based on the Create_Drivepack.dat input file:

```
arkc -create -drivepack -ICreate_Drivepack.dat
```

- priority(Optional)
  The parameter priority assigns a priority to a drive contained in a Drivepack. This allows an administrator to define precisely which drive will receive first the data from a back-up operation. For instance, if drive A has a priority of 1 and drive B has a priority of 5, the A drive will be used first for a backup operation. If this operation requires more drives, then drive B will be used, followed by any other drives in the order of their priority.

  This parameter can accept values from 1 to 128, with a value of 1 (highest priority) being assigned by default.

  Combining the priority parameter with the nbdrive offers a high level of

flexibility for the operation of a group of drives. Please refer to the above information for more examples on how to add this parameter to a Drivepack input file.

Drivepack deletion

The correct syntax to delete a Drivepack is the following one:

```
arkc -drivepack -delete -[D|I] <parameters...>
```

Where the parameters can be the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the Drivepack to be deleted. For instance:

  ```
  arkc -drivepack -delete -D name=DLT_drivepack
  ```

  A list of all the Drivepacks available on a given Arkeia server can be obtained with the command:

  ```
  arkc -drivepack -list
  ```

- `dkid`(Optional)
  The `dkid` parameter is the ID assigned to the Drivepack when it was created by Arkeia. This ID value can be used instead of the `name` parameter described above. For instance, the two commands below are equivalent, assuming that `3bc46702` is the correct ID of the Drivepack named `DLT_drivepack`:

  ```
  arkc -drivepack -delete -D name=DLT_drivepack
  arkc -drivepack -delete -D plid=3bc46702
  ```

Drivepack modification

To modify a Drivepack using the `modify` command of `arkc`, the following syntax should be respected:

```
arkc -drivepack -modify -D name=name_of_drivepack <parameters>
```

It is highly recommended to use an input file (with the option `-I`) to modify a Drivepack. Each item to be added to a Drivepack should be contained within an `<ITEM>...</ITEM>` statement. The parameters that can be added to such an input file are the following ones:

- `drvname`(Optional)
  The parameter `drvname` contains the name of the drive to be added to the Drivepack. You can obtain a list of drives installed on the Arkeia server by entering the following command:

  ```
  arkc -drive -list
  ```

  To add a drive to a Drivepack, the following syntax should therefore be used:

```
<ITEM>
drvname=DLT01
</ITEM>
```

The above <ITEM> will add the drive DLT01 to a Drivepack, assuming a drive of that name exists on the Arkeia server.

- drvid(Optional)
  The parameter drvid contains the ID of the drive to be added to the Drivepack. This parameter is used in much the same way than the parameter drvname above. For instance, the following <ITEM> statement will add the drive DLT01 to a Drivepack, assuming 3bc419c3 is the correct ID of this drive:

  ```
  <ITEM>
  drvid=3bc419c3
  </ITEM>
  ```

- priority(Optional)
  The parameter priority defines the priority level assigned to the drive. This parameter can accept values from 1 to 128, with the default (and highest possible) value being 1. Please note that this parameter can only be used within an input file, and not on the command line.

The following parameters can also be modified using the -modify command of arkc: name, dkid (Drivepack name and ID) and nbdrive (number of drives to be used in the Drivepack). Since these are functionally equivalent to the parameters of the create command, please refer to *The create command of arkc* on page *15* for more information.

Listing available drivepacks with the list commands

It is possible to display detailed information on the Drivepacks with the list command. For more information on this command, please refer to the chapter of this manual titled: *The informational commands and objects of arkc* on page *48*.

17. The library object

This chapter will detail all the arkc commands that can be applied to the library object.

Library creation

Arkeia is able to manage complex backup hardware, such as large tape libraries. This type of hardware should be defined within Arkeia in order to be used for back-up and restore operations.

The standard syntax to create a library with arkc is the following:

`arkc -library -create -[D|I<input file>] <parameters...>`

The parameters that can be accepted by arkc while creating a library are the following ones:

- `name`**(<u>Required</u>)**
  The parameter `name` assigns a descriptive name to the library being created. This parameter is mandatory. For instance:

  ```
  name=DLT_Lib04
  ```
  or:

  ```
  name=[Accounting DLT library]
  ```

- `type`**(<u>Required</u>)**
  The parameter `type` defines the type of library being created. Please note that the value of `type` must be entered precisely as shown. Some possible values for this parameter are:

  ```
  ADIC100-LTO
  ATL-L200
  AUTOPAK
  BLACKJACK_X_21
  DLT4700
  DLT7
  DLTSTOR114
  EXB-17D7
  EXB-60
  EXB-EZ17
  EXB-X200
  FALCON-1530
  FALCON-1560
  FASTSTOR
  FILE
  HP19-LTO
  HP660-LTO
  HP660
  HPC1191
  HPDAT6
  Harrier8150
  Harrier830
  Harrier850
  IBM7336
  IBM7337
  ITL2225
  ITL4115
  ITL7115
  LIBRA16
  LIBRA18
  LIBRA8
  LP1117
  MAGSTAR_L12
  MAGSTAR_L18
  MAGSTAR_L24
  MAGSTAR_L32
  SONY_DMSB9
  ```

```
STD4586NP-12
STD4586NP
STK7430
STK9710
STK9714
SUN-L1000
SUN-L700-9840
SUN-L700-DLT
SUN-L9
TSL-A300
TSL-S7000
TWINLIBRA16
TWINLIBRA18
```

You can obtain a complete list of all the hardware supported by Arkeia by typing the command: `arkc -library -type` on a command line. For more information, please see chapter 6: The informational commands of arkc.

Please note: The value of `type` should be entered exactly as shown in the list above. These values are case-sensitive. Therefore, the value `hp19-dlt` and `HP19-DLT` are not equivalent.

A library creation requires the appropriate license of Arkeia. If you do not have a license for a library, `arkc` will produce an error message and exit. For more information on the Arkeia licensing scheme, please contact: `sales@arkeia.com`.

- `libdev`(**Required**)
  The parameter `libdev` indicates which UNIX device is the controller of the library being created. This device will vary according to the system and its configuration. For instance, here are two possible values for `libdev`:

  `libdev=/dev/sg0`(Linux)
  `libdev=/dev/rsst4`(Sun Solaris)

  For more information on control devices, please refer to the manual of your UNIX/Linux system, and to the output of the `dmesg` UNIX command.

- `comment`(Optional)
  The parameter `comment` assigns a descriptive comment to the library being created. This parameter is useful to display more information on a given library. For instance, the following is a valid comment for `arkc`:

  ```
  comment=[DLT library for Accounting Department]
  ```

  Please note that, as you can see above, comments that contain spaces should be included between square brackets: `[...]`.

Library deletion

The correct syntax to delete an Arkeia library is the following one:

```
arkc -library -delete -[D|I] <parameters...>
```

Where the parameters can be the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the library to be deleted. For instance:

  ```
  arkc -library -delete -D name=Lib04
  ```

  A list of all the libraries available on a given Arkeia server can be obtained with the command:

  ```
  arkc -library -list
  ```

- `libid`(Optional)
  The `libid` parameter is the ID assigned to the library when it was created by Arkeia. This ID value can be used instead of the `name` parameter described above. For instance, the two commands below are equivalent, assuming that `3cb46702` is the correct ID of the Library named `Lib04`:

  ```
  arkc -library -delete -D name=Lib04
  arkc -library -delete -D libid=3cb46702
  ```

Library modification

To modify an existing library, please refer to the library creation described in *Library creation* on page *29* of this manual or to the section of the same title above. Please note that the correct syntax is the following:

```
arkc -library -modify -[D|I<input file>] <parameters...>
```

While modifying a library, only the following parameters can be used: `name`, `libid` (Library name and ID), `libdev` (control device for the library) and `comment`. For more information on these parameters, please refer to the section named *Library creation* on page *110*.

Listing available libraries with the list command

The arkc list command provides detailed information on the different libraries that are available on a give server. For more information on this command, please refer to *The informational commands and objects of arkc* on page *48*.

Listing the drives of a library with the drvlist command

For the `library` object, arkc offers an additional `list` command available, which is the `drvlist` command.

While the list command, when applied to the library object, will list all the available libraries connected to a Arkeia server, the drvlist command will list the drives that are installed in a specific library. The syntax of this command is the following:

```
arkc -library -drvlist -[D|I] <parameters...>
```

The parameters that can be applied to this command are the following ones:

- name(**Required**)
  The name parameter contains the name that was assigned to the library when it was created. Please note that this parameter is required by the drvlist command. For example:

  ```
  arkc -library -drvlist -D name=ArkeiaLib01
  ```

  The above command will list all the tape drives installed within the library named ArkeiaLib01.

- libid(Optional)
  The libid parameter is the ID that was assigned to the library when it was created. It can replace the name parameter. For instance, the two commands below are equivalent, assuming that 3b614c30 is the valid ID for library Autopack30:

  ```
  arkc -library -drvlist -D name=Autopack30
  arkc -library -drvlist -D libid=3b614c30
  ```

The output of the drvlist command can be modified with the help of the following filters:

- drvnameDisplays only the names of the drives present in a library.
- drvidDisplays only the IDs of the drives present in a library.
- drvnumDisplays only the numbers of the drives present in a library.

Please note it is possible to combine several filters. For instance, the following command would list the drives present in library IBMlib and display both their names and IDs:

```
arkc -library -drvlist -D name=IBMlib -Fdrvname -Fdrvid
```

Additional library information with the type command

The type command is designed to provide information on the type of objects that can, for instance, be created by arkc. Its syntax is as simple as the one of the list command:

```
arkc -object -type
```

We have seen several examples in the previous sections of this manual. For libraries, the command arkc -library -type will list all the libraries that can be managed by Arkeia.

Starting and stopping a library

The start and stop commands can be used to manage the tape libraries.

Starting a library

In order to use a given library, Arkeia has to start a specialized module, that allows it to *drive* (meaning: use all the functions of) a library. This module is usually launched automatically, for instance when a backup is started. However, it is sometimes necessary to start a library manually. To do this, the following syntax should be respected:

```
arkc -library -start -[D|I<input file>] <parameters...>
```

This command accepts only one parameter, which is:

- name(**Required**)
  The name parameter contains the name of the tape library to be started. This name is assigned to the tape library during its creation. For instance:

  ```
  arkc -library -start -D name=[HP SureStore 1/9 DLT]
  ```

  The command above would start the tape library named HP SureStore 1/9 DLT. Please note the use of the square brackets around the name of the library.

Please note: it is possible to replace the name parameter by the libid parameter. The libid parameter is the ID of the library that should be started. It can be obtained by entering the command:

```
arkc -library -list -D name=name_of_the_library | more
```

Therefore, the following start command is correct, assuming 3af012b1 is the correct ID for the library HP SureStore 1/9 DLT:

```
arkc -library -start -D libid=3af012b1
```

Stopping a library

It is possible to stop a library manually by using the stop command of arkc. For instance:

```
arkc -library -stop -[D|I<input file>] <parameters...>
```

This command accepts the same parameters as the start command. For instance, the following commands are valid:

```
arkc -library -stop -D name=[HP SureStore 1/9 DLT]
arkc -library -stop -D libid=3af012b1
```

Additional tape-related commands for tape libraries

There are four arkc tape-related commands that are specifically designed for tape libraries. These commands are:

- settapeLibrary slots and tapes configuration and assignement.

- `unsettape`Modification of the configuration of the library slots/tape.
- `load`Load a tape from a pre-defined slot.
- `unload`unload a tape from a pre-defined slot.

The rest of this section is going to detail these commands and their usage.

Assigning logical tapes to library slot with settape

For a library to function properly under Arkeia, it is necessary to assign existing tapes to the different slots present in the library. This is done through the settape command. The syntax of this command is the following:

```
arkc -library -settape -[D|I<input file>] <parameters...>
```

The possible parameters for this command are the following:

- `name/libid`(**Required**)
  The `name` and `libid` parameters contain the name and the ID of the library, respectively. This parameter is required to indicate which library is going to be configured by the rest of the `settape` command. For instance, the following parameters are equivalent, assuming `3b614c30` is the correct ID of the library named `Autopack30`:

  ```
  name=Autopack30
  libid=3b614c30
  ```

  Or, for a complete command:

  ```
  arkc -library -settape -D name=Autopack30 ...
  ```

- `tpname/tpid`(**Required**)
  The `tpname` and `tpid` parameters contain the name and the ID of the tape that should be assigned to a slot in the library. Only one of these parameters need to be set for the `settape` to be carried out correctly. For instance, the following command would assign the tape named `VVXX01` to a slot in the library named `Autopack30`:

  ```
  arkc -library -settape -D name=Autopack30 tpname=VVXX01
  ```

- `slot`(**Required**)
  The `slot` parameter contain the number of the slot the tape defined above should be assigned to. This parameter only accepts numerical values, ranging from `1` (first slot) to the last slot available in the library. For more information on the number of slots available, please refer to the documentation supplied with the library hardware. For instance, the following parameter would assign a tape to slot number `5`:

  ```
  slot=5
  ```

Removing logical tapes from library slots with unsettape

The unsettape command provides a way to remove (un-assign) logical tapes from Arkeia tape libraries. This command has the following syntax:

```
arkc -library -unsettape -[D|I<input file>] <parameters...>
```

This command accepts the following parameters:

- `name/libid`(**Required**)
  The `name` and `libid` parameters contain the name and the ID of the library, respectively. This parameter is required to indicate which library is going to be targeted by the rest of the `unsettape` command. For instance, the following parameters are equivalent, assuming `3b614c30` is the correct ID of the library named `Autopack30`:

  ```
  name=Autopack30
  libid=3b614c30
  ```

  Or, for a complete command:

  ```
  arkc -library -unsettape -D name=Autopack30 ...
  ```

- `slot`(**Required**)
  The `slot` parameter contain the number of the slot that should not contain a tape. This parameter only accepts numerical values, ranging from `1` (first slot) to the last slot available in the library. For more information on the number of slots available, please refer to the documentation supplied with the library hardware. For instance, the following parameter would unassign the tape of slot number `8`:

  ```
  slot=8
  ```

Loading a tape from a slot with load

Once a tape has been set to a slot, it is possible to *load* (transfer) the tape from its slot into one of the tape drives of the library. This is done with the `load` command. The syntax of this command is the following:

```
arkc -library -load -[D|I<input file>] <parameters...>
```

The parameters that can accepted by the load command are the following ones:

- `name/libid`(**Required**)
  The `name` and `libid` parameters contain the name and the ID of the library, respectively. This parameter is required to indicate which library is going to be affected by the `load` command. For instance, the following parameters are equivalent, assuming `3b614c30` is the correct ID of the library named `Autopack30`:

  ```
  name=Autopack30
  libid=3b614c30
  ```

Or, for a complete command:

```
arkc -library -load -D name=Autopack30 ...
```

- slot(**Required**)
  The slot parameter contain the number of the slot that should contain a tape. This parameter only accepts numerical values, ranging from 1 (first slot) to the last slot available in the library. For more information on the number of slots available, please refer to the documentation supplied with the library hardware. For instance, the following parameter would load the tape of slot number 8:

  ```
  slot=8
  ```

- drvnum(Optional)
  The drvnum is the logical number of the drive that should receive the tape from the slot. For instance, the following parameter would load a tape into the drive number 6 of a tape library:

  ```
  drvnum=6
  ```

  Please note that, by default, the load command transfers the tape into the first drive.

Removing a tape from a slot with unload

Once a tape has transferred to a tape drive, it is possible to *unload* (transfer) the tape back to its slot from one of the tape drives of the library. This is done with the unload command of arkc. The syntax of this command is the following:

```
arkc -library -unload -[D|I<input file>] <parameters...>
```

The parameters that can accepted by the unload command are the following ones:

- name/libid(**Required**)
  The name and libid parameters contain the name and the ID of the library, respectively. This parameter is required to indicate which library is going to be affected by the unload command. For instance, the following parameters are equivalent, assuming 3b614c30 is the correct ID of the library named Autopack30:

  ```
  name=Autopack30
  libid=3b614c30
  ```

  Or, for a complete command:

  ```
  arkc -library -unload -D name=Autopack30 ...
  ```

- slot(**Required**)
  The slot parameter contain the number of the slot that should receive the unloaded tape. This parameter only accepts numerical values, ranging from 1

(first slot) to the last slot available in the library. For more information on the number of slots available, please refer to the documentation supplied with the library hardware. For instance, the following parameter would unload the tape back to slot number 8:

```
slot=8
```

- drvnum(Optional)
  The drvnum is the logical number of the drive that should eject the tape back to the slot. For instance, the following parameter would eject the tape contained into drive number 6:

  ```
  drvnum=6
  ```

Additional drive-related commands for tape libraries

There are two arkc commands that provide additional, drive-related, functionality for tape libraries. These commands are:

- attachAttach additional tape drives to a library
- detachRemove additional tape drives from a library

The rest of this section will detail the syntax and usage of these commands:

Attaching additional drives to a tape library with attach

The attach command allows the library to be re-configured to attach additional drives to it. Please note that this command cannot be used if the target tape library has only one drive. The syntax for this command is:

```
arkc -library -attach -[D|I<input file>] <parameters...>
```

The valid parameters for this command are:

- name/libid(**Required**)
  The name and the libid parameters contain, respectively, the name and the ID of the library that will be re-configured by the attach command. Please note that only one parameter needs to be set for the attach command to work properly. For instance, the following parameters are equivalent, assuming 3b614c30 is the correct ID of the library named Autopack30:

  ```
  name=Autopack30
  libid=3b614c30
  ```

- drvname/drvid(**Required**)
  The drvname and drvid parameters contain, respectively, the name and the ID of the drive that will be attached to a tape library. Please note that only one parameter needs to be set for the attach command to work properly. For instance, the following parameters are equivalent, assuming 3bc419c3 is the correct ID of the tape drive named DLT01:

```
drvname=DLT01
drvid=3bc419c3
```

- drvnum(Optional)
  The drvnum is the logical number that will be assigned to the drive defined by the parameter drvname or drvid (see above for more information on these parameters. For instance, the following parameter would assign the number 6 to the drive attached to a tape library:

  ```
  drvnum=6
  ```

Please note: Assigning the wrong logical number to a drive attached to a library can result in severe problems. In case of a doubt, remove all the drives attached to a library and re-attach them in a correct order, paying attention to the logical numbers that are assigned to each.

Removing drives from a library with detach

The detach command allows the drives previously attached to library to be removed from it. The syntax of this command is the following:

```
arkc -library -detach -[D|I<input file>] <parameters...>
```

The parameters that are accepted by htis command are the following:

- name/libid(**Required**)
  The name and the libid parameters contain, respectively, the name and the ID of the library that will be re-configured by the detach command. Please note that only one parameter needs to be set for the detach command to work properly. For instance, the following parameters are equivalent, assuming 3b614c30 is the correct ID of the library named Autopack30:

  ```
  name=Autopack30
  libid=3b614c30
  ```

- drvname/drvid(**Required**)
  The drvname and drvid parameters contain, respectively, the name and the ID of the drive that will be detached from the tape library. Please note that only one parameter needs to be set for the detach command to work properly. For instance, the following parameters are equivalent, assuming 3bc419c3 is the correct ID of the tape drive named DLT01:

  ```
  drvname=DLT01
  drvid=3bc419c3
  ```

  Please note that is is possible to use several names of IDs on the command line to detach several drives with only one command. For instance:

```
arkc -library -detach -D name=Lib04 drvname=DLT1
drvname=DLT2 ...
```

18. The backup object

This chapter of the arkc manual will detail the commands that can be applied to the backup object.

Starting a backup

The start command can be used to launch a backup with arkc. The standard syntax for this command is the following one:

```
arkc -backup -start -[D|I] <parameters...>
```

The following parameters can be applied to this command:

- skname(**Required**)
  The skname parameter contains the name of the Savepack to be saved by the backup. This name is assigned to the Savepack during its creation. For instance:

  ```
  arkc -backup -start -D skname=JackHome
  ```

  The command above would start the backup of the Savepack named Jack-Home.

- plname(**Required**)
  The plname parameter contains the name of the tape pool to be used by the backup. This name is assigned to the tape pool during its creation. For instance:

  ```
  arkc -backup -start -D skname=JackHome plname=Pool01
  ```

  The command above would start the backup of the Savepack named Jack-Home, on the tape pool named: Pool01.

- dkname(**Required**)
  The dkname parameter contains the name of the Drivepack to be used for the backup. This name is assigned to the Drivepack during its creation. For instance:

  ```
  arkc -backup -start -D skname=JackHome plname=Pool01
  dkname=DSK_01
  ```

The command above would start the backup of JackHome, on the tape pool Pool01 and on the Drivepack named DSK_01. Please note that these options should all be contained on the same line.

Please note: The parameters skname, plname and dkname, while they must be entered on the command line to start a backup can be replaced by the parameters skid, plid and dkid. These parameters are respectively the IDs for the Savepack, the tape pool and the Drivepack to be used.

The following optional parameters can be used to start a backup with arkc:

- policy(Optional)
  The parameter policy determines the tape policy to be used for the backup. This parameter can accept two values: COMPLETE or NEW. When using the COMPLETE value, Arkeia will always complete (fill up) the tape which has been used. If the NEW value is used, Arkeia will always use a new tape when starting a backup. By default, policy has the value COMPLETE.

- wait(Optional)
  The parameter wait determines the behavior of arkc once a backup is launched with arkc -backup -start. This parameter can accept two values: YES or NO. If the value YES is used, arkc will wait until the end of the backup to return, which means the user will not be able to enter any new command until the completion of the backup. If the value NO is used, arkc will return immediately and allow the user to enter other commands.

- email(Optional)
  The parameter email determines if Arkeia sends an email once the backup is finished, or if the backup requires a new tape to be completed. This parameter can accept two values YES and NO. If the value YES is used, Arkeia will send an email to the user, indicating if the backup is finished and/or if the software requires an additional tape to complete the backup operation. If the value NO is used, then Arkeia will not send an email. The default value of email is YES.

Please note: In order to receive an email, the parameter email, defined for a user, must contain a valid email address. Even if the email parameter of a backup is defined, it will not send any information if the user email parameter does not contain any information.

- mode(Optional)
  The parameter mode controls the operational mode of Arkeia for a backup. Two modes (and values of mode) are available: CONTINUOUS and STANDARD. When the STANDARD mode of operation is chosen, the backup will stop once it is completed, and will be considered as terminated by the Arkeia server.

When the CONTINUOUS mode is selected, the backup task will not be considered as finished once it has saved all the files in the Savepack, and more Savepacks can be added to the backup task. By default, the value of mode is STANDARD: if you are unsure which mode is best suited to your need, please use the default.

- comment(Optional)
  The parameter comment is used to add a more descriptive comment to a backup. For instance:

  ```
  comment=[Complete user data backup, including email dir]
  ```

  Please note that, as seen above, comments that contain spaces should be enclosed within square brackets [...].

- retention(Optional)
  The parameter retention determines the duration of the data retention of the backup. This parameter must be completed by the retunit parameter (see below) and can accept values starting at 1. For instance:

  ```
  retention=6
  ```

- retunit(Optional)
  The parameter retunit determines the type of unit used by the retention parameter. It can accept the following values: DAY, WEEK, MONTH and YEAR. For instance, to take back the example above:

  ```
  retention=6
  retunit=MONTH
  ```

  This defines a retention period of 6 months for the data of the backup being started with arkc.

- parallelism(Optional)
  The parameter parallelism determines the number of concurrent data flows that will be used for the backup. This parameter can accept numerical values from 1 to 128. By default, Arkeia uses the maximum number of flows available for its backups.

- type(Optional)
  The parameter type determines the type of backup that will be performed by Arkeia. This parameter can accept three values: TOTAL, INCREMENTAL and ARCHIVE. By default, the value of type is TOTAL. For more information about

the different type of backup available under Arkeia, please refer to the Arkeia User Manual.

- based_on_bksid(Optional)
  The based_on_bksid parameter is valid only if the type parameter is set to INCREMENTAL. In this case, the based_on_bksid parameter should contain the ID of the previous backup the new backup is based on.

- based_on_date(Optional)
  The based_on_date parameter is valid only if the type parameter is set to INCREMENTAL. In this case, the based_on_date parameter should contain the date of the previous backup the new backup is based on.

Modifying a running backup with the connect command

The standard syntax to modify a running backup with the connect command is the following:

```
arkc -backup -connect -[D|I<input file>] <parameters...>
```

The following parameters can be used to modify a running backup job:

- bksid(**Required**)
  The bksid parameter contains the ID of the backup to be modified. This ID is returned by arkc when a backup is started with the start command.

- wait(Optional)
  The wait parameter is used to determine the behavior of arkc at the end of the backup operation. This parameter can accept two values: YES and NO. If the value YES is used, arkc will only accept new commands from the user after the backup is completed. If the value NO is used, arkc will immediately accept new commands from the user.

- exec(Optional)
  The exec parameter modifies the behavior of the running backup. The following operation can be entered as the value of this parameter:

  ABORTStops the current backup.
  ADDSKAdd a new Savepack to the current backup.
  SETTAPEIndicates that a new tape has been inserted.
  SETMODEModifies the mode of the backup.

  For instance, the following command would abort the running backup with the ID 3a6718dc:

```
arkc -backup -connect -D bksid=3a6718dc exec=ABORT
```

- drvname(Optional)
  The parameter drvname determines the drive that contains the new tape. This parameter is **required** if the exec parameter has the value SETTAPE. Please refer to the exec parameter, above, for more information. For instance:

  ```
  arkc -backup -connect -D bksid=3a6718dc exec=SETTAPE
  drvname=ecrix_2
  ```

  The command above indicates to the running backup with an ID of 3a6718dc that a new tape has been inserted in the drive named ecrix_2 and can be used to save the data being backed-up. Please note that this command should be placed all on one line.

- mode(Optional)
  The parameter mode should only be used if the exec parameter is set to SET-MODE. There are two possible values for mode, which are: STANDARD (the job will be stopped once the backup is finished) and CONTINUOUS (the job will stay active even after the end of the backup).

- skname(Optional)
  The parameter skname should only be used if the exec parameter is set to ADDSK (add a Savepack). The value of skname should then be the name of the Savepack to be added to the running job. For instance:

  ```
  arkc -backup -connect -D bksid=3a6718dc exec=ADDSK
  skname=JackFiles
  ```

  The command above adds the Savepack named JackFiles to the running backup designated by the ID: 3a6718dc. Please refer to the exec parameter above for more information.

Displaying the completed backups with done

It is possible to display a list of all backup operations completed by using the done command of arkc.

The syntax of this command is the following one:

```
arkc -backup -done [-[D|I<input file>] <parameters...>]
```

The command arkc -backup -done will simply list all the backup that have been completed. To obtain a more informational message, it is necessary to filter the output of this command with the following filters:

- `bksid`Displays only the ID of the backups.
- `date`Displays only the date (numerical) of the backups.
- `sdate`Displays only the date (string) of the backups.

For instance, here are a few examples of filters, and their outputs:

```
$ arkc -backup -done
bksid=3a6718dc

$ arkc -backup -done -Fsdate
sdate=2001/01/18 16:25

$ arkc -backup -done -Fsdate -Fbksid
bksid=3a6718dc
sdate=2001/01/18 16:25
```

Displaying the list of backups in progress with running

The `running` command can be used to display the list of backups that are in progress on the Arkeia server. Further information about a specific job can then be queried using the `status` command (see below).

The standard syntax for the `running` command is the following:

```
arkc -[backup|restore] -running
```

For instance, to display a list of all backup operations currently in progress, it is possible to enter:

```
arkc -backup -running
```

This command will then display the IDs of all the backup in progress on the Arkeia server. To obtain more information on a specific job, please refer to the `status` command below.

Displaying information on a running backup with status

To obtain detailed information on a running backup, the `status` command should be used. This command has the following syntax:

```
arkc -backup -status -[D|I<input file>] <parameters...>
```

This command accepts only one parameter, which is detailed below:

- `bksid`(**Required**)
  The `bksid` parameter contains the ID of the backup whose detailed status should be displayed. This ID is displayed when a backup is started with the `start` command. For more information, please refer to the next chapter of this manual: *The start and the stop commands of arkc* on page *63*.

For instance, the following command:

```
arkc -backup -status -D bksid=3bcc2793
```

will return the possible values:

- `0`The backup job is running.

- 1The backup needs a new tape.
- 2The backup is now finished.

If the value returned is 1, the status command also returns two values, which are:

- drvnameName of the drive to be used for the new tape.
- tpnameName of the new tape to be inserted in the drive.

Please note that it is possible to modify the behavior of a running backup by using the connect command to modify its parameters. For more information on the connect command, please refer to *Modifying a job with the connect command* on page *73*.

19. The restore object

This chapter will detail all the arkc commands that can be applied to the restore object.

Starting a restoration

The command start allows you to start a restoration operation from the command line. The standard syntax for this command is:

```
arkc -restore -start -[D|I<input file>] <parameters...>
```

The parameters that can be accepted by arkc with this command are the following ones:

- file(**Required**)
  The file parameter must be inserted on the command line or in an input file. This parameter contains the name of the file (or directory) that should be restored. For instance:

  ```
  arkc -restore -start -D
  file=stormbringer:/home/jack/docs/AnnualReport_2000.ps
  ```

  The arkc command above, if entered on a command line, will start the restoration of the file named AnnualReport_2000.ps, which can be found on the machine named stormbringer in directory: /home/jack/docs/. Please note that the command above should be entered on a single line.

- from/to(Optional)
  The parameter from allows the restoration to take place with a redirection. This means that a file restored from a directory X can be restored to another directory Y. For instance, taking back the example above:

  ```
  arkc -restore -start -Ireports.txt
  ```

(Content of reports.txt:)

```
# Example of restoration __________

file=AnnualReport_2000.ps
from=babylon:/home/jack/files/Reports/
to=babylon:/home/jack/tmp/arkeia/temporary/
```

In the example above the file AnnualReport_2000.ps, whose original directory was /home/jack/files/Reports on the workstation babylon, (from= line) will be restored instead to the directory: /home/jack/tmp/arkeia/temporary (to= line). This allows the user to check the content of the restored file before moving it back to its original position.

Please note: A from parameter should always be followed and completed with a to parameter. If this is not the case, the restoration will be rejected by arkc with an error message.

- wait(Optional)
  The parameter wait determines the behavior of arkc once a restoration is launched with arkc -restore -start. This parameter can accept two values: YES or NO. If the value YES is used, arkc will wait until the end of the restoration to return, which means the user will not be able to enter any new command until the completion of the operation. If the value NO is used, arkc will return immediately and allow the user to enter other commands on the command line.

- drvname(Optional)
  The parameter drvname contains the name of the drive to be used for the restoration, in a single-drive environment. This name is assigned during the creation of the drive. For instance:

  ```
  drvname=SonyDLT
  ```

  On a single-drive system, the parameter above indicates that the drive named SonyDLT should be used for the restoration operation.

Please note: It is possible to replace the drvname parameter by the drvid parameter. The drvid should then contain the ID of the drive to be used for the restoration, in the case of a single-drive configuration. This ID is assigned to the drive during its creation and can be obtained by entering the following command:

```
arkc -drive -list -D name=name_of_the_drive
```

- `rstopt`(Optional)
  The `rstopt` parameter determines the different options that should be applied to the files that will be managed by the restoration. This parameter accepts the following values, that will influence the restoration:

  `REST_TIMES`Restore the time information of the files.
  `RST_PERMS`Restore the access rights of the files.
  `RST_BYID`Restore by user identifier.
  `RST_BYNAME`Restore by user name.
  `RST_CHKTIME`Check if the existing file (if any) is newer.

Modifying a restoration with the connect command

The standard syntax to modify a running restoration with the `connect` command is the following:

```
arkc -restore -connect -[D|I<input file>] <parameters...>
```

The following parameters can be used to modify a running restoration:

- `bksid`(**Required**)
  The `bksid` parameter contains the ID of the restoration to be modified. This ID is returned by `arkc` when a restoration is started with the `start` command.

- `wait`(Optional)
  The `wait` parameter is used to determine the behavior of `arkc` at the end of the restoration. This parameter can accept two values: `YES` and `NO`. If the value `YES` is used, `arkc` will only accept new commands from the user after the restoration is completed. If the value `NO` is used, `arkc` will immediately accept new commands from the user.

- `exec`(Optional)
  The `exec` parameter modifies the behavior of the restoration. The following operation can be entered as the value of this parameter:

  `ABORT`Stops the current restoration.
  `SETTAPE`Indicates that a new tape has been inserted.
  `SETDRIVE`Set a drive for the current restoration.

  For instance, the following command would abort the running restoration with the ID `3a6729dc`:

  ```
  arkc -restore -connect -D rstid=3a6729dc exec=ABORT
  ```

- drvname(Optional)
  The parameter drvname indicates either (a) the drive that contains the new tape (if exec has the value SETTAPE) or (b) the drive that should be used for the restoration (if exec has the value SETDRIVE). This parameter is **required** if the exec parameter has the value SETTAPE or SETDRIVE. Please refer to the exec parameter, above, for more information. For instance:

  ```
  arkc -restore -connect -D rstsid=3a6729dc exec=SETDRIVE
  drvname=ecrix_2
  ```

  The command above indicates to the restoration with an ID of 3a6729dc that the drive to be used for the operation is the ecrix_2 drive. Please note that this command should be placed all on one line.

Displaying the list of restorations with running

The running command can be used to display the list restoration that are in progress on the Arkeia server. Further information about a specific job can then be queried using the status command (see below).

The standard syntax for the running command is the following:

```
arkc -restore -running
```

For instance, to display a list of all backup operations currently in progress, it is possible to enter the above command, and it will then display the IDs of all the backup in progress on the Arkeia server. To obtain more information on a specific job, please refer to the status command below.

Displaying the state of a restoration with status

To obtain detailed information on a running restoration, the status command should be used. This command has the following syntax:

```
arkc -restore -status -[D|I<input file>] <parameters...>
```

This command accepts only one parameter, which is detailed below:

- rstid(**Required**)
  The rstid parameter contains the ID of the restoration whose detailed status should be displayed. This ID is displayed when a restoration is started with the start command. For more information, please refer to the *The start and the stop commands of arkc* on page *63*.

For instance, the following command:

```
arkc -restore -status -D rstid=3bcc4545
```

will return the possible values:

- 0The restoration is in progress.
- 1The restoration needs a new tape.
- 2The restoration is now completed.
- 3The restoration requires a new drive.

If the value returned is 1, the status command also displays two values, which are:

- drvnameName of the drive to be used for the new tape.
- tpnameName of the new tape to be inserted in the drive.

If the value returned is 3, the only value displayed will be drvname, as the restoration only requires a new drive, and not a new drive and a new tape.

Please note that it is possible to modify the behavior of a restoration in progress by using the connect command to modify its parameters. For more information on the connect command and its uses, please refer to *Modifying a job with the connect command* on page *73*.

Displaying restoration-related log information

To display restoration-related information contained in the Arkeia log, the journal object should be used. The correct syntax for this object is the following:

```
arkc -journal -jrestore -[D|I<input file>] <parameters...>
```

For more information on the journal object and its parameters, please refer to the section named *Displaying log information with the journal object* on page *60*, in the chapter named *The informational commands and objects of arkc* on page *48* of this manual.

20. The file object

This chapter of the arkc manual will explain the different commands that can be applied to the file object.

Listing files present on a tape with the list command

The object file is used to obtain a list of files contained on a given tape. When using the command arkc -file -list, it is therefore necessary to supply the name of the tape to be listed with the tpname parameter. The standard syntax is therefore:

```
arkc -file -list -D tpname=name_of_tape
```

For instance:

```
arkc -file -list -D tpname=DLT_140
```

The command above will list all the files contained on the tape named DLT_140. For more information on the file object, please refer to the second part of this manual.

It is possible to filter the output of -file -list command with the following filter:

- fileDisplays only the files present on the tape.

Finding a file with the where command

Finding a file that should be restored can be a time-consuming process. The `where` command can help in this process. This command only applies to the `file` object.

The correct syntax for this command is the following:

```
arkc -file -where -[D|I<input file>] <parameters...>
```

The command `where` searches the internal Arkeia database and tries to find a match to the search criteria that were given.

This command accepts the following parameter:

- `file`(**Required**)
  The `file` parameter contains the name of the file (or directory) that should be searched for. The search criteria should be in the following format:

  ```
  machine:directory/directory/file
  ```

  For instance:

  ```
  arkc -file -where -D file=pluto:/home
  ```

  The command above will start a search in the Arkeia database for the directory `/home` that was backed-up from the machine named `pluto`.

The output of the `where` command can be modified with the following filters:

- `tpname`Displays only the name of the tape(s) where the files are found.
- `tpid`Displays only the ID of the tape(s) where the files are found.

21. The user object

This chapter will detail the different `arkc` commands that can be applied to the `user` object.

User creation

Creating a user can be accomplished with the `create` command with the following syntax:

```
arkc -user -create -[D|I] <parameters>
```

The parameters that can be used to create a user are the following ones:

- `name`(**Required**)
  The parameter `name` assigns a descriptive name to the user being created. The correct syntax for this parameter is, for example:

  ```
  name=John
  ```

If a name must contain spaces, it should be enclosed between square brackets: `[...]`. For instance:

```
name=[John Smith]
```

- `node`(**Required**)
  The parameter `node` contains the name of the Arkeia client the user will interact with. This parameter has the following syntax:

  ```
  node=stormbringer
  ```

  In the example above, the name of the Arkeia client which is to be used is `stormbringer`. A node should always be assigned to a newly-created user.

- `email`(Optional)
  The `email` parameter is used to define an email address that Arkeia can use to warn the user of the completion of an operation. The email address should be a standard address, that can be reached by the Arkeia server. For instance:
  `email=john.smith@foobar.com`

- `passwd`(Optional)
  The `passwd` parameter defines the password used by the user to connect to the Arkeia server defined by the `node` parameter. It is highly recommended to use secure UNIX passwords to avoid security problems. For instance:

  ```
  passwd=DF56AX82VV08
  vpasswd=DF56AX82VV08
  ```

Please note: the password defined by the passwd parameter should always be confirmed with the vpasswd parameter, as in the example above.

User deletion

The correct syntax to delete an Arkeia user is:

```
arkc -user -delete -[D|I] [name=User_Name]
```

Where the parameters can be the following ones:

- `name`(**Required**)
  The `name` parameter should contain the name of the user to be deleted. For instance:

```
arkc -user -delete -D name=John
```

A list of all the users available on a given Arkeia server can be obtained with the command:

```
arkc -user -list
```

User modification

To modify an Arkeia user, the parameters to be used are the same as with the `create` command. For more information, please refer to the section *User creation* on page *132*. The valid syntax is therefore:

```
arkc -user -modify -[D|I] <parameters>
```

The parameters used to modify a user are `name`, `role`, `node`, `email`, `passwd` and `vpasswd`. For more information on these, please refer to the section above.

Listing available users with list

It is possible to list all the valid users for a given Arkeia server by using the `list` command. To do this simply enter:

```
arkc -user -list
```

For more information on the `list` command, please refer to the section named *Listing objects with the list command* on page *49* of this manual.

The role command

The `role` command is a limited version of the `type` command, that applies only to the `user` objects of Arkeia. To obtain all the roles available to Arkeia users, enter the command:

```
arkc -user -role
```

The three roles that are available for Arkeia users are: `OPERATOR`, `USER` and `ADMINISTRATOR`. Please note that, when a user is created, a role should be defined and entered in upper case, under the parameter `role`. For instance, the following input file defines a valid Arkeia user:

```
# ----- File name jack.txt -----
# User creation file for arkc.
# -----------------------------
# Basic features

name=[Jack Smith]
role=ADMINISTRATOR

node=station01

# Optional features
email=jack@foobar.com

# End of file.
```

To create the user `Jack Smith`, the following command should therefore be entered, assuming the input file is named `jack.txt`:

```
arkc -user -create -Ijack.txt
```

22. The tape duplication object

This chapter will detail the arkc commands that can be applied to the tape duplication object.

Important information:

Please note that the tape duplication functions are only available in version 5.x (and following) of arkc.

Previous versions of arkc do not offer the tape duplication func-

Contrary to a lot of arkc objects, the tape duplication is started automatically, and cannot be created, deleted or modified.

The normal syntax of the tape duplication is the following:

```
arkc -duplication -start -D <parameters...>
```

The -D parameter indicates to arkc that its arguments and parameters are entered on the command line.

The following parameters must be entered on the command line, when duplicating a tape from one drive to another:

- src_tape_name or src_tapeid: (**Required**)

  These parameters are required by arkc.

  These parameters should contain either the name or the ID of the source tape. For instance, the following names and IDs are considered valid, and will be accepted by arkc for a tape duplication command:

  ```
  src_tape_name=DLT004
  src_tapeid=3bc1d2e2
  ```

  As can be seen above, the src_tape_name may be easier to remember, but both these values are valid, and accepted, by arkc. To display a list of available source tapes, enter the command: arkc -tape -list (without the quotes). The tape name is assigned when a tape is created by Arkeia.

- src_drv_name or src_drvid: (**Required**)

  These parameters are required by arkc.

These parameters should contain either the name or the ID of the source drive. For instance, the following names and IDs are considered valid, and will be accepted by `arkc` for a tape duplication command:

```
src_drv_name=RD_LTO
src_drvid=3a530bcf
```

As can be seen above, the `src_drv_name` may be easier to remember, but both these values are valid, and accepted, by `arkc`. To display a list of available drives, enter the command: `arkc -drive -list` (without the quotes).

- `dst_drv_name` or `dst_drvid`: (**Required**)

  These parameters are required by `arkc`.

  These parameters should contain either the name or the ID of the destination drive. This drive must be compatible with the source drive! For instance, the following names and IDs are considered valid, and will be accepted by `arkc` for a tape duplication command:

  ```
  dst_drv_name=RD_LTO
  dst_drvid=3a530bcf
  ```

  As can be seen above, the `dst_drv_name` may be easier to remember, but both these values are valid, and accepted, by `arkc`. To display a list of available drives, enter the command: `arkc -drive -list` (without the quotes).

Additionally, the following parameters can be entered, when duplicating a tape that is contained in a tape library:

- `dst_lib_name` or `dst_libid`: (**Required**)

  These parameters are required when using `arkc`, if duplicating in a library.

  These parameters should contain either the name or the ID of the destination library. This library must contain the destination drive defined above! For instance, the following names and IDs are considered valid, and will be accepted by `arkc` for a tape duplication command:

  ```
  dst_lib_name=RD_LTO
  dst_libid=3a530bcf
  ```

  As can be seen above, the `dst_lib_name` may be easier to remember, but both these values are valid, and accepted, by `arkc`. To display a list of available libraries, enter the command: `arkc -library -list` (without the quotes). Please note that defining `dst_lib_name` or `dst_libid` means that `dst_slot` (see below) must be defined as well.

- `dst_slot`: (**Required**)

This parameter is required when using `arkc`, but only if duplicating in a library.

This parameter should contain the number of the slot that contains the destination tape. This parameter must be defined if duplicating to a library! For instance, the following is considered as valid value, and will be accepted by `arkc` for a tape duplication command:

```
dst_slot=6
```

This slot must contain the tape that will be used as a target by the duplication operation! Please note that, when duplicating from/to a library, it is not necessary to define a source slot for the source tape, as Arkeia is able to retrieve the correct tape based on its ID and/or name. Also, please note that defining `dst_slot` means that `dst_lib_name` or `dst_libid` (see above) must be defined as well.