

**ND-WORM Services**

**No Duplicate-Write Once Read Many Services**

---

# **ND-WORM Client/Server**

---

Prototype User's Manual 1.1

July 2000 revision

<b>1</b>	<b>INTRODUCTION .....</b>	<b>3</b>
<b>2</b>	<b>NOMENCLATURE.....</b>	<b>3</b>
<b>3</b>	<b>ND-WORM CONFIGURATION AND SETUP INFORMATION .....</b>	<b>5</b>
3.1	FINDING THE CONFIGURATION FILE .....	5
3.1.1	<i>How do clients find their configuration file.....</i>	5
3.1.2	<i>How does the wormserver find the configuration file.....</i>	6
3.1.3	<i>How does the wormserver find the section to use within the configuration file.....</i>	6
3.1.4	<i>How does a client find the instance of the wormserver to connect to.....</i>	6
3.2	FORMAT OF THE CONFIGURATION FILE .....	6
3.2.1	<i>Server configuration file.....</i>	6
3.2.2	<i>Client configuration file.....</i>	8
3.3	CONFIGURATION FILE REFERENCE .....	8
3.3.1	<i>Database-connection related configuration: needed by the worm server .....</i>	8
3.3.2	<i>Error Logging configuration information: needed by the worm server and all worm clients 9</i>	9
3.3.3	<i>Transient naming service (rmiregistry) configuration information.....</i>	9
3.3.4	<i>Other Optional configuration information. ....</i>	9
<b>4</b>	<b>THE TRANSIENT NAMING SERVICE (RMIREGISTRY).....</b>	<b>10</b>
4.1	STARTING RMIREGISTRY .....	10
<b>5</b>	<b>THE ND-WORM SERVER .....</b>	<b>10</b>
5.1	STARTING THE SERVER .....	10
<b>6</b>	<b>CLIENT TOOLS .....</b>	<b>11</b>
6.1	QUICK REFERENCE .....	11
6.2	DETAILED CLIENT TOOLS REFERENCE .....	11
6.2.1	<i>winsert .....</i>	11
6.2.2	<i>wsearch .....</i>	14
6.2.3	<i>wretrieve .....</i>	17
6.2.4	<i>wretrievec.....</i>	20
6.2.5	<i>wlist.....</i>	22
6.2.6	<i>wdelete .....</i>	23
6.2.7	<i>wsync .....</i>	23
<b>7</b>	<b>CHANGES SINCE LAST RELEASE.....</b>	<b>24</b>



# 1 Introduction

The ND-WORM system is a client server application that supports the concept of a file system based archive. The archive is to be used mainly for storing files in a safe area. The ND-WORM acronym means Write Once Read Many – Non Duplicate. Duplicates are not allowed in the WORM. A file is a duplicate of a file already in the WORM if it has the same digest as the worm file.

The ND-WORM can't be used in a standalone manner. It relies on a back end Informix database (for now). The files are copied in the worm area under a unique name, which is the MD5 file digest (32 characters). Special care should be exercised when backing up the database, since the files in the worm are useless without the database catalog.

The ND-WORM suite consists of a server, a transient naming service (the Sun Java SDK rmiregistry) and a set of command line client tools used to send server requests.

**Delivery notes:** the ND-WORM deliverable consists of a gzipped tar file worm1.0.tar.gzip. Unzip and untar the file in the area where the package will be installed. For further details about the delivery contents and installation procedures please read the README file in the installation root directory.

## 2 Nomenclature

Following is a brief description of some terms which we use extensively in the ND-WORM system. The explanations below should clarify the context in which these terms are used.

<b>Word</b>	<b>Our Meaning</b>
<b>File</b>	Entity that is being archived.
<b>Digest</b>	32 characters string that uniquely identifies the file. The MD5 algorithm is used to compute the digest. Digests are a way to guarantee data integrity – but we are using them to uniquely identify a data stream (a file in our case).
<b>Pool</b>	A hierarchical way to organize a set of correlated file. It translates in a subdirectory under the WORM root area. Pools are read from the server configuration file and created on disk and in the database.
<b>Pool Name</b>	A unique way to identify a pool (a pool alias).
<b>Pool Path</b>	The subdirectory path under the WORM root area.
<b>Set</b>	A group of tightly correlated filed, that can be archived and retrieved through one operation.
<b>Set Alias</b>	A unique way to identify a set. A set alias is generated automatically at file insertion time if the user does not provide one.
<b>Keywords</b>	A space separated set of words that can be used to find a file or a set of files. The keywords are associated with the appropriate entities at insertion time.
<b>User id</b>	The user id of the client account that performed the insert operation.
<b>Retention period</b>	The number of days that the files (set of files) will be retained in the archive.
<b>Internal id</b>	The internal id is a unique way to identify a WORM entity. It is a database-generated integer.
<b>Root path</b>	The paths under which all the files managed by an instance of the server are stored (in their appropriate pools).
<b>Default path</b>	The path where all files without a pool name specification will be stored.
<b>Retrieve conflicts</b>	Anomalies that might appear during a retrieve operation when multiple file names are associated to the same digest or when one name appear under multiple digests. The conflicts can be resolved by specifying the unique id of the entity that needs to be retrieved, rather than the attributes that create conflicts.



## 3 ND-WORM Configuration and Setup Information

This section describes the configuration of the ND-WORM system components, and how you can configure them to meet your needs.

The ND-WORM consists of three basic modules:

1. A wormserver that manages the WORM area and services client requests.
2. A set of command line client tools that perform basic WORM operations. They are winsert, wretrieve and wdelete. More client and admin tools will be developed..
3. A transient naming service that allows clients to connect to the appropriate instances of the WORM server. Transient means that if the naming service is started, the server have to be restarted in order to be found by the client. A transient naming service does not support persistence.

The server and client programs read setup information from the configuration file. Different amounts of information are needed by the client and server programs.

The client only needs to know how to contact the transient naming service and how to log information in a log file.

The server also needs to know how to contact the transient naming service in order to register itself, how to log information, and how to contact the back end Informix database.

The ND-WORM configuration has been designed to make it easy for new users to attach and run. The following environment variables have to be set in order to provide a basic configuration.

The WORM server needs:

`$JAVAHOME` – path to the JDK 1.2  
`$WORM_HOME` – path to the home directory where the ND-WORM system has been installed  
`$WORM_CONFIG_FILE` – path to the server configuration file

The WORM client tools need:

`$JAVAHOME` - path to the JDK 1.2  
`$WORM_HOME` - path to the home directory where the ND-WORM system has been installed  
`$WCLIENT_CONFIG_FILE` – path to the server configuration file

ND-WORM Administrators will have to read this section carefully to see how to set up all the needed configuration information.

The server configuration file is divided into sections. Each section contains key=value pairs that define a single instance of the ND-WORM server. This way, you can have multiple instances of the ND-WORM server running, but you only have to manage a single configuration file.

### 3.1 Finding the configuration file

#### 3.1.1 How do clients find their configuration file

The ND-WORM client tools will examine the contents of the `WCLIENT_CONFIG_FILE` for the complete path and filename of the client configuration file. If the environment variable is not set or errors are found in the configuration file, the client tools will abort.

### 3.1.2 How does the wormserver find the configuration file

The ND-WORM server will examine the contents of the **WORM\_CONFIG\_FILE** for the complete path and filename of the server configuration file. If the environment variable is not set or errors are found in the configuration file, the client tools will abort. Additional information about the reason for failure will be displayed.

### 3.1.3 How does the wormserver find the section to use within the configuration file

The wormserver startup script accepts one command line parameter, which is the server mode. The mode must match exactly the name of a server configuration file section.

*Example:*

```
[SAMPLE] is the name of the section in the configuration file
wormserver SAMPLE is the way to start a server running in SAMPLE mode
```

### 3.1.4 How does a client find the instance of the wormserver to connect to

In order for a client to connect to the appropriate server, the mode in which the server is running needs to be known to the client. A client configuration file entry called **WORMSERVERMODE** needs to be set to the mode in which the server is running in.

*Example:*

```
WORMSERVERMODE=SAMPLE
```

Currently there is no way to specify the wormserver mode on the client command line, so different configuration files should be used by clients that connect to different servers. The configuration file to use should be controlled by pointing the **WCLIENT\_CONFIG\_FILE** variable to the desired configuration file.

## 3.2 Format of the configuration file

### 3.2.1 Server configuration file

Here is an example of a server configuration file that defines one WORM server instance, illustrated by the section labeled SAMPLE:

```
#####
## THIS IS A SAMPLE UNIX CONFIG FILE FOR A SERVER RUNNING IN SAMPLE MODE ##
## Look for the XXX in the comments above the lines that have to be      ##
## customized.                                                            ##
#####
[SAMPLE]
#
# Information for connecting to the database:
#
# Password utilized for connecting XXX
PASSWORD=xxxxxxx
# Username utilized for connecting XXX
USERNAME=xxxxxxx
# DATABASE JDBC Driver class name
DATABASEDRIVER="com.informix.jdbc.IfxDriver"
```

```

# JDBC syntax URL, replace with the appropriate values XXX
DATABASEURL="jdbc:informix-
sql://informixHostname:informixPortNo/databaseName:INFORMIXSERVER=informixServerInstance
"
# Information related to connection pooling:
#
# Number of connections permanently acquired at startup from
# the DbServer On release, the pool connections are returned
# to the pool
OPENDBCONNECTIONS=7
# Extra connections that can be acquired if all pool
# connections are allocated On release the non pool
# connections are returned to the DbServer
MAXDBCONNECTIONS=9
#
# Information related to logging.
#
# file where error and informational messages are logged.
# UNIX style separator for unix environments XXX
MESSAGELOG="//home//username//logs//WormServerInstanceName.log"
# A descriptive string to put in the log file, in front of
# every message.
UNIT="WormServer"
# How much stuff to print out. Only two values right now:
# MaxDebug, NoDebug
DEBUGSTATUS="MaxDebug"
#
# Information on how to connect to the RMI Registry XXX
#
RMIREGISTRYPORT=1099
# this should be the RMI registry host IP, start by using the same host where the
# wormserver will be running
RMIREGISTRYHOST=x.x.x.x
# POOL INFORMATION
# roo path - under which all other paths will be created
# SYSTEM SPECIFIC SEPARATORS
# this is the root path, under which all other pools will be created XXX
ROOT_PATH=/home/username/archive_home
# this is a directory path that will be created under the root path and will
# contain all files archive without specifying a specific pool XXX
DEFAULT_PATH=default_path
# description of all other pools, the POOLx is just a counter, the stuff
# on the right side is a generic pool name that needs to be unique and an
# associated path XXX
POOL1="POOL1NAME:pool1_underroot//pool1_path"
POOL2="POOL2NAME:pool2_underroot//pool2_path"
POOL3="POOL3NAME:pool3_underroot//pool3_path"
POOLx="POOLxNAME:poolx_path"
# this is the digest algorithms being used
UNIQUE_FILENAME_GENERATOR=MD5
# block size in K, that is used for the file copy operation
BLOCK_SIZE=256

```

Additional instances can be added by providing new sections with customized information. You can use any instance names you like. You can have as many instance sections as you like. The instance names and keywords are case-sensitive, so be sure to reference them properly when using them. Right now you cannot attach more than one section name to the same set of key=value pairs.

Comment lines in the configuration file start with the # character and go to the end of the line. Comments must be on their own line. You should surround the values with quoted whenever special characters need to appear in the value. The special characters are: "/", "\", ";", ":", ".".

### 3.2.2 Client configuration file

Here is an example of the minimal configuration file that a WORM client tool needs:

```
#####
## this is a sample client config file for a UNIX CLIENT ##
## Look for the XXX in the comments above the lines that ##
## have to be customized. ##
#####
# the port where the RMI registry listens on XXX
RMIREGISTRYPORT=1099
# this host ip where the RMI registry is running XXX
RMIREGISTRYHOST=x.x.x.x
# worm server mode, determines the server instance that the client
# will connect to XXX
WORMSERVERMODE=SAMPLE
# client log file
# file where error and informational messages are logged.
# all worm clients (winsert, wretrieve, wsearch) are logging info in
# this file XXX
# UNIX style separator for unix environments
MESSAGELOG="/home/username/worm/logs/wclient.log"
# A descriptive string to put in the log file, in front of
# every message.
UNIT="wclient"
# How much stuff to print out. Only two values right now:
# MaxDebug, NoDebug
DEBUGSTATUS="MaxDebug"
```

### 3.3 Configuration File Reference

The configuration keywords support a variety of purposes, so we document them here in several sections.

#### 3.3.1 Database-connection related configuration: needed by the worm server

<i>Keyword</i>	<i>Purpose/Meaning</i>
PASSWORD	Password to use to connect to the database.
USERNAME	Username to use when connecting to the database. This does not have to be the userid under which the wormserver is running.
DATABASEDRIVER	Vendor specific syntax that allows a JDBC client to connect to the database. By making it a configuration item we would allow the server to connect to different vendors via their JDBC drivers.
OPENDBCONNECTIONS	Number of continuously open database connections to maintain between the wormserver and the database server.
MAXDBCONNECTIONS	Maximum number of database connections to maintain between the wormserver and the database. Be sure that MAXDBCONNECTIONS >= OPENDBCONNECTIONS. The extra connections are opened on an 'as needed' basis, and are closed right away.

### 3.3.2 Error Logging configuration information: needed by the worm server and all worm clients

This information is needed both by the worm client tools wormserver.

The paths for the files where state and debug information is written by ESSW modules can be full paths or relative paths to the directory in which the programs are started.

<i>Keyword</i>	<i>Purpose/Meaning</i>
MESSAGELOG	Log file location for the server and/or client tools.
UNIT	A simple name to put at the beginning of each log message. Usually this is just the name of the application. For example: wormserver, wormclienttools. This is helpful in figuring out what program produced a particular log file.
DEBUGSTATUS	Level of logging to produce. There are currently two possible values to use here: "NoDebug" or "MaxDebug". The NoDebug does not produce any debug information, the MaxDebug logs verbose debug information.

Note: If problems are encountered before the log files can be opened, then error information will be logged in the following location:

/var/tmp/ESSW\_ERROR.LOG.<youruserid>

### 3.3.3 Transient naming service (rmiregistry) configuration information

The rmiregistry needs to be started before the server. The following configuration items allow the server to locate the naming service for registering. The WORM client tools will use the naming service to get the server information for the server they need to connect to.

<i>Keyword</i>	<i>Purpose/Meaning</i>
RMIREGISTRYHOST	Hostname where the rmiregistry is running. Note: this keyword is used by both the server and the client tools.
RMIREGISTRYPORT	Port on which the rmiregistry is listening for requests. Note: this keyword is used by both the server and the client tools.

### 3.3.4 Other Optional configuration information.

These are documented here for completeness. Most of them should be left to the defaulted values:

<i>Keyword</i>	<i>Purpose/Meaning</i>
ROOT_PATH	Top level area where the WORM archive is located. Only used by the server.
DEFAULT_PATH	Area under the root path where files are stored when no pool is specified on insert.
POOLX	Where X = (1..n). Designator for a pool. Used by the server. The value contains a unique pool name and a

	pool path. For example: "POOL1NAME:pool1_underroot//pool1_path"
BLOCK_SIZE	The block size in kilobytes used by the server for file copying. It default to 256. Used only by the server.
UNIQUE_FILENAME_GENERATOR	The algorithm used to compute the digest. It should be MD5. Used only by the server.
WORMSERVERMODE	The mode of the server to which the client needs to connect. Used only by the client tools.

## 4 The transient naming service (rmiregistry)

The transient naming service **rmiregistry** is a server-side naming repository that allows remote clients to get a reference to the remote server object.

Before you start the RMI Registry, make sure the shell or window in which you run the **rmiregistry** command does not have a **CLASSPATH** environment variable that points to the **WORM** classes anywhere on your system. If the RMI Registry finds these classes when it starts, it will not load them from the server-side Java VM, which will create problems when clients try to download the remote server classes.

### 4.1 Starting rmiregistry

The **rmiregistry** requires the **JAVAHOME** environment variable pointing to the directory where JDK 1.2 or later is installed.

The **rmiregistry** runs on the default 1099 port. You can specify a different port by adding the port number as follows:

```
$WORM_HOME/rmiregistry 4444 &
```

The hostname and port number that **rmiregistry** listens on will be the entries used by the **wormserver** and the entire set of worm client tools.

## 5 The ND-WORM server

The Write Once Read Many – Non Duplicate server services all the client requests (currently coming from the client command line tools). It copies files or sets of files to the archive, catalogs them and sends pertinent information back to the client who displays it. It also can search the catalog and retrieve files or sets of files from the archive.

### 5.1 Starting the server

The script that starts the **wormserver** executable is found in **\$WORM\_HOME/bin/wormserver**.

To run the **wormserver**, you will need to:

1. have **\$WORM\_HOME** set properly to the **WORM** home directory.
2. have **\$JAVAHOME** set properly to the directory where JDK 1.2 has been installed.
3. have **\$JAVAHOME/bin** in your path, since **wormserver** is a java 1.2 program.
4. have **\$WORM\_HOME/bin** in your path, that would allow you to run the other **WORM** executables.



**Description:**

Client tool used to insert one or multiple files to the WORM archive. At insertion the user can specify: the pool name where the file will be stored, the retention period, keywords to associate to the files themselves etc.

Since the files will be stored under different names in the WORM archive, the user can create a link to the WORM copy of the file.

A set of files that are closely related can be inserted as a set, thus allowing their retrieval as a single entity.

The actual filename of the WORM file is a 32-character server generated filename. The original filenames are also stored because they are needed for the archive retrieval operations.

**Options:**

**-f** Files to be inserted in the archive. One or more file names can be provided. If multiple file names are used, the **-s** option can be utilized to generate a "set id" that will allow the retrieval of all files through a single operation.

**-p** Pool name where the file(s) should be archived. If no pool name is provided, the default pool defined in the server configuration file will be used.

**-r** Retention value in days. It will default to "forever" that will allow the file to remain in the WORM until the **wdelete** command will be issued on the file.

**-k** space separated keywords that could be used for searching this file(s).

**-l** flag to specify that the only output from the insert command should be list of fully qualified path and filename of all files archived with the **-f** option. The output will contain the server generated digest as the filename and can be used to create a link to the file that has been archived.

**-s** Set id will be generated. Should be used if multiple files are archived through one operation and they need to be retrieved through one operation. Tightly correlated files can be archived with this option, the generated set id can be used for retrieval. The id will be printed on stdout at the completion of the **winsert** command. A set alias can be provided in order to easily identify the set for a retrieve operation. If no alias is provided, an alias will be automatically generated.

**-help** displays the command usage.

**Examples:**

**A.** `winsert -f file3.tmp file4.tmp -p POOL1NAME -k file3file4`  
Configuration file: D:\worm\bin\worm.config.client.local  
\*\*\* winsert \*\*\* Completed SUCCESSFUL INSERT  
\*\*\* COPIED TO ARCHIVE \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file3.tmp, 167ba4cdeba7c9a3c017d3f0fd155a29, 2000-05-15 15:49:03.428, 1)  
(file4.tmp, 1aebaad1b3cb6d3fd195ea4ffe221034, 2000-05-15 15:49:03.868, 2)

*Archives the files file3.tmp and file4.tmp from the current directory in the WORM pool named POOL1NAME and the associated keyword being file3file4.*

*The command output contains the file original name, the generated digest, the archival timestamp and the generated unique id.*

**B.** `winsert -f file3.tmp file4.tmp -p POOL1NAME -k reinsert`  
Configuration file: D:\worm\bin\worm.config.client.local  
\*\*\* winsert \*\*\* Completed SUCCESSFUL INSERT

```
*** ALREADY IN ARCHIVE ***
(NAME, DIGEST, INSERT DATE, ID)
(file3.tmp, 167ba4cdeba7c9a3c017d3f0fd155a29, 2000-05-15 15:54:12.141, 3)
(file4.tmp, 1aebaad1b3cb6d3fd195ea4ffe221034, 2000-05-15 15:54:12.232, 4)
```

*Repeats the operation in step A. Notice the output stating that the files were already in the archive. This is an example of the ND feature of the WORM. No copy was performed – but the insert operation was cataloged.*

```
C. winsert -f d:\file1.tmp -k same name different digest
Configuration file: D:\worm\bin\worm.config.client.local
*** winsert *** Completed SUCCESSFUL INSERT
*** COPIED TO ARCHIVE ***
(NAME, DIGEST, INSERT DATE, ID)
(file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)
```

*Archives the file d:\file1.tmp and associates the specified keywords with the file. In this example another file1.tmp already exists in the WORM archive with a different digest. The current file is copied to the archive.*

```
D. winsert -f file1.tmp.a.txt -p POOL3NAME -k digestconfl
Configuration file: D:\worm\bin\worm.config.client.local
*** winsert *** Completed SUCCESSFUL INSERT
*** ALREADY IN ARCHIVE ***
(NAME, DIGEST, INSERT DATE, ID)
(file1.tmp.a.txt, df223a55ce9e6c358204abdce7af31df, 2000-05-15
16:07:43.098, 8)
```

*An example of archiving a file that exists in the archive, having the same digest but a different original name. Since the digest is used for WORM uniqueness, the WORM copy will not take place – but the insert operation is cataloged.*

```
E. winsert -f file5.tmp file6.tmp -p POOL2NAME -s -k file5file6s
Configuration file: D:\worm\bin\worm.config.client.local
*** winsert *** Completed SUCCESSFUL INSERT
SETID: 1 SETALIAS: alias1
*** COPIED TO ARCHIVE ***
(NAME, DIGEST, INSERT DATE, ID)
(file5.tmp, 60d412ffad106b77542f238a2f5b7adf, 2000-05-15 16:12:01.018, 9)
(file6.tmp, 7485502e08fe41f354f9a6761b16ca83, 2000-05-15 16:12:01.359, 10)
```

*Archives the files file5.tmp and file6.tmp AS A SET from the current directory in the WORM pool named POOL2NAME and the associated keyword being file5file6s. Since no set alias has been provided, a unique alias is generated. The set will also be associated with the specified keywords.*

*The command output contains the generated set id, set alias, file original name, the generated digest, the archival timestamp and the generated unique id.*

```
F. winsert -f file7.tmp file8.tmp -p POOL3NAME -s set78alias -k file7file8sa
Configuration file: D:\worm\bin\worm.config.client.local
*** winsert *** Completed SUCCESSFUL INSERT
SETID: 2 SETALIAS: set78alias
*** COPIED TO ARCHIVE ***
(NAME, DIGEST, INSERT DATE, ID)
```

(file7.tmp, b8028db8268065c6c59511326b5088bb, 2000-05-15 16:16:48.852, 11)  
(file8.tmp, e18f4c7fd255c350ce32222e2fb2e4bb, 2000-05-15 16:16:48.952, 12)

*Archives the files file7.tmp and file8.tmp AS A SET from the current directory in the WORM pool named POOL3NAME and the associated keyword being file7file8sa. The provided UNIQUE set alias will be used. The set will also be associated with the specified keywords.*

*The command output contains the generated set id, the specified set alias, file original name, the generated digest, the archival timestamp and the generated unique id.*

**G.** winsert -f file10.tmp file11.tmp -s set78alias  
Configuration file: D:\worm\bin\worm.config.client.local  
\*\*\* winsert \*\*\* Completed SUCCESSFUL INSERT  
SETID: 2 SETALIAS: set78alias  
\*\*\* COPIED TO ARCHIVE \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file10.tmp, c32563ac090d1c7c25b169312b74c0c4, 2000-05-15 16:21:51.808, 13)  
(file11.tmp, 4eb16eeaf5300e2ecfd4c3446508f35c, 2000-05-15 16:21:51.878, 14)

*Assuming the operation in step F took place before, this command associates two other files to the set with the set78alias. The keywords associated to the set are overwritten.*

**H.** winsert -f file14.tmp file15.tmp -s set1415 -l  
Configuration file: D:\worm\bin\worm.config.client.local  
\*\*\* winsert \*\*\* Completed SUCCESSFUL INSERT  
SETID: 3 SETALIAS: set1415  
\*\*\* COPIED TO ARCHIVE \*\*\*  
(LINK PATH)  
(D:\arhive\_home\default\_path\c4480262c3b62e11c2303b8c57fd0f4b)  
(D:\arhive\_home\default\_path\3fa3ce3ac6ac21d2725fcc497375fe87)

*Archives the files file14.tmp and file15.tmp AS A SET from the current directory in the DEFAULT WORM pool, with set1415 as the associated keyword. The provided UNIQUE set alias will be used. The set will also be associated with the specified keywords. The -l flag controls the output format. Instead of the NAME, DIGEST, INSERT DATE and INTERNAL ID the path to the archive file is displayed.*

## 6.2.2 wsearch

Searches for files or sets of files in the WORM system and displays the result on stdout.

**wsearch**            [-f] filename  
                      [-i] internal id  
                      [-d] digest  
                      [-k] keyword1 keyword2 ...  
                      [-s] [set alias]  
                      [-u] user id  
                      [-v]

**Description:**

The output will be a list that displays all (or selected) metadata about all file(s) or set(s) matching the specified criteria.

**Options:**

- f** Original file name that has been provided on the winsert operation. If used in conjunction with the **-v** flag, all file information will be displayed, otherwise only the digest, original name, timestamp and internal id will be displayed, which should be sufficient to uniquely identify the file, in order to use it in wretrieve operations.
- i** Internal id of the file being searched. Cannot be used in conjunction with any other options, except **-v** (verbosity).
- d** File digest that the search should be based on. Cannot be use din conjunction with any other options, except **-v** (verbosity).
- k** Keywords that the search should be based on. One or multiple keywords can be used. If the **-s** flag is present a set keyword search will be performed, otherwise a file keyword search will be performed.
- s** Indicates a set based search. Sets matching the specified criteria, rather than all files in the set will be returned. Tightly correlated files can be located with this option, and operated on as a set - rather than using individual commands. An optional set alias can follow and the search for the specified set will be performed. This option can be used in conjunction with the **-k** and **-v** options.
- u** User id of the user that performed the winsert command. All files inserted by the indicated user id will be returned. Can be used in conjunction with the **-v** option.
- v** Verbosity flag. If not used, only information that uniquely identifies a set or a file is displayed. If present, all metadata about the search results will be displayed.

**Examples:**

- A.** `wsearch -f file1.tmp`  
 Configuration file: d:\worm\bin\worm.config.client.local  
 \*\*\*\* SEARCH RESULTS \*\*\*\*  
 (NAME, DIGEST, INSERT DATE, ID)  
 (file1.tmp, df223a55ce9e6c358204abdce7af31df, 2000-05-15 16:00:33.049, 5)  
 (file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)
- Searches for all files that were stored with the name file1.tmp in all the WORM pools, starting with the default, and displays the non-verbose output.*
- B.** `D:\worm\test>d:\worm\bin\wsearch -i 5`  
 Configuration file: d:\worm\bin\worm.config.client.local  
 \*\*\*\* SEARCH RESULTS \*\*\*\*  
 (NAME, DIGEST, INSERT DATE, ID)  
 (file1.tmp, df223a55ce9e6c358204abdce7af31df, 2000-05-15 16:00:33.049, 5)
- Searches for the file having -1 as its internal id, and displays the non-verbose output.*
- C.** `D:\worm\test>d:\worm\bin\wsearch -s set78alias`  
 Configuration file: d:\worm\bin\worm.config.client.local  
 (SET ID, SET ALIAS)  
 (2, set78alias)

*Searches for the set of files that has been stored under the set78alias. Knowing a valid set alias, all files stored in that set can be retrieved via one operation.*

- D.** D:\worm\test>d:\worm\bin\wsearch -u username  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\*\* SEARCH RESULTS \*\*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file4.tmp, 1aebaad1b3cb6d3fd195ea4ffe221034, 2000-05-15 15:49:03.868, 2)  
(file3.tmp, 167ba4cdeba7c9a3c017d3f0fd155a29, 2000-05-15 15:49:03.428, 1)  
(file4.tmp, 1aebaad1b3cb6d3fd195ea4ffe221034, 2000-05-15 15:54:12.232, 4)

*Searches for all files that were archived by the user "username", and displays the non-verbose output.*

- E.** D:\worm\test>d:\worm\bin\wsearch -d 1aebaad1b3cb6d3fd195ea4ffe221034  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\*\* SEARCH RESULTS \*\*\*\*  
NAME, DIGEST, INSERT DATE, ID)  
(file4.tmp, 1aebaad1b3cb6d3fd195ea4ffe221034, 2000-05-15 15:49:03.868, 2)  
(file4.tmp, 1aebaad1b3cb6d3fd195ea4ffe221034, 2000-05-15 15:54:12.232, 4)

*Searches for all files in the archive that have the specified digest, and displays the non-verbose output.*

- F.** D:\worm\test>d:\worm\bin\wsearch -k file3file4  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\*\* SEARCH RESULTS \*\*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file4.tmp, 1aebaad1b3cb6d3fd195ea4ffe221034, 2000-05-15 15:49:03.868, 2)  
(file3.tmp, 167ba4cdeba7c9a3c017d3f0fd155a29, 2000-05-15 15:49:03.428, 1)

*Searches for all files in the archive that have the specified associated keywords and displays the non-verbose output.*

- I.** D:\worm\test>d:\worm\bin\wsearch -k file7file8sa -s  
Configuration file: d:\worm\bin\worm.config.client.local  
(SET ID, SET ALIAS)  
(2, set78alias)

*Performs a set keyword search, for all sets having the specified associated keywords, and displays the non-verbose output.*

- J.** D:\worm\test>d:\worm\bin\wsearch -k file3file4 -v  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\*\* SEARCH RESULTS \*\*\*\*  
(NAME, DIGEST, INSERT DATE, ID, OWNER, ORIGINAL LOCATION, RDAY, KEYWORDS, POOL PATH, XML)  
(file4.tmp, 1aebaad1b3cb6d3fd195ea4ffe221034, 2000-05-15 15:49:03.868, 2, username, D:\worm\test, 12000, file3file4, D:\arhive\_home\pool1\_underroot\pool1\_path, null)  
(file3.tmp, 167ba4cdeba7c9a3c017d3f0fd155a29, 2000-05-15 15:49:03.428, 1, username, D:\worm\test, 12000, file3file4, D:\arhive\_home\pool1\_underroot\pool1\_path, null)

*Performs a file keyword search, for all files having the specified associated keywords, and displays the verbose output.*

## 6.2.3 wretrieve

Retrieves one or multiple files from the WORM system.

```
wretrieve    -f file1 file2 file3 ...
             [-i] fileId1, fileId2, fileId3 ...
             [-s] set alias
             [-r] directory to retrieve the files into
             [-l]
             [-help]
```

### Description:

Client tool used to retrieve one or multiple files from the WORM archive. Can copy the files into the client indicated area or can return the file path that can be used by the client to create a link. Currently the server must have RW permissions to the retrieve directory.

If conflicts exist (multiple file names for a given digest or the same file name under 2 digests) the part of the request that is unambiguous will succeed and information to help solve the conflicts will be displayed. If multiple versions of one file exist, they need to be narrowed down to one version, by using the file internal id instead of its name. This information can be obtained by using the **wsearch** client tool.

A set of files that were archived as a set, can also be retrieved as a set, thus avoiding the need of repeating the wretrieve command for each member of the set.

The original file name will be used to copy the file out of the archive, since the server generated file name is meaningless for the user.

### Options:

- f** Files to be retrieved from the archive. One or more file names can be provided. Possible scenarios:
- A. If only one file exists with a given name it will be retrieved in the current directory or the retrieve directory specified by the -r option. Appropriate access in the retrieve directory needs to be provided.
  - B. If multiple files with the same name exist in the WORM, they will have different digests, so a conflict will occur. The information that will uniquely identify each file in the conflict will be displayed. This scenario covers the situation in which multiple versions of the same file exist in the WORM.
- i** one or multiple Ids for the files to be retrieved from the archive. Specifying internal IDs is optional. A valid entity id can be obtained by using a wsearch operation based on file metadata (keywords, file name, owner, etc). The internal ID is a cleaner way to uniquely identify a file in the WORM. The actual uniqueness is guaranteed by the timestamp together with the file digest.
- t** timestamp for the file to be retrieved. Specifying a timestamp is optional. A valid timestamp can be obtained by using a wsearch operation based on file metadata (keywords, file name, owner, etc). The timestamp together with the file digest (specified with the -d option) uniquely identify a file in the WORM.
- d** digest for the file to be retrieved. Specifying a digest is optional. A valid digest can be obtained by using a wsearch operation based on file metadata (keywords, file name, owner, etc). Can be used individually or in conjunction with the -t flag. MD5 or SHA algorithms are used to compute the digest.

**-p** Pool name where the file(s) were archived. If no pool name is provided, the default pool defined in the server configuration file will be first used to search for the desired file(s). If that fails, a search based on the filename will be initiated. Using a pool name might speed up the catalog search operations.

**-s** Set id to be used for retrieval. It allows files that have been archived as one unit (a set) to be retrieved the same way - as opposed to issuing an individual command for each member of the set. It can be used on retrieval only if the same flag has been used on archival, when multiple files were archived through one operation. Tightly correlated files can be archived and retrieved with this option. Cannot be used together with the **-l** option, since the links have individual target names that would have to be specified.

**-a** Set alias to be used for retrieval. It allows files that have been archived as one unit (a set) to be retrieved the same way - as opposed to issuing an individual command for each member of the set. It can be used on retrieval only if an alias has been provided on archival, when multiple files were archived through one operation. Tightly correlated files can be archived and retrieved with this option. Cannot be used together with the **-l** option, since the links have individual target names that would have to be specified. If multiple sets have the same alias the command will fail and one unique set will have to be selected and retrieved using the set id flag.

**-r** Fully qualified output directory where the files will be retrieved. The wormserver daemon needs to have read/write permissions to that area. If no output directory is specified, the current client directory will be used.

**-l** Link name that should be created to the desired file. This option should be used when the user wants to avoid the overhead of a copy operation out of the WORM system. If this option is used, the file will not be copied to the retrieve directory (or the current directory if no **-r** flag is used), a link to the WORM file will be created there (or maybe a file which contains the link creation commands will be created). This option works only if there aren't multiple versions of the specified file in the worm. In order to uniquely qualify the file to link to the **-d** and **-t** options can be used.

**-v** Verification flag. The server sends back the original file digest and after retrieval, if a file was copied out of the WORM, the server digest will be compared to the client calculated digest.

#### Examples:

**A.** D:\worm\test>D:\worm\bin\wretrieve -f file1.tmp -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\* wretrieve RETRIEVE CONFLICTS \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)  
(file1.tmp, df223a55ce9e6c358204abdce7af31df, 2000-05-15 16:00:33.049, 5)

*Retrieves the file that was stored under the original name file1.tmp. In this scenario, the WORM archive contains two different digests for file1.tmp. That means that the same file has multiple versions that have been archived. The information that would help the user solve the conflict will be displayed. The client can retrieve the desired versions based on the internal id.*

**B.** D:\worm\test>D:\worm\bin\wretrieve -i 6 -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\* wretrieve RETRIEVED FILES \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)

*Retrieves the file having 6 as an internal id from the WORM archive. The output contains minimal information about the file that was retrieved. Since the -l option was not present the file was actually copied to the retrieve directory.*

C. D:\worm\test>D:\worm\bin\wretrieve -i 6 -r d:\archiveRetrieve -l  
Configuration file: d:\worm\bin\worm.config.client.local  
(NAME, WORM PATH)  
(file1.tmp, D:\arhive\_home\default\_path\33e8def83e25c9f2dad53db52783d8e3)

*Retrieves the archive path for the file having 6 as an internal id. The output contains the WORM path together with the original file name for the desired file. The path together with the original file names can be used to create links to the archived file.*

D. D:\worm\test>D:\worm\bin\wretrieve -s set78alias -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local

```
*** wretrieve RETRIEVED FILES ***  
(NAME, DIGEST, INSERT DATE, ID)  
(file7.tmp, b8028db8268065c6c59511326b5088bb, 2000-05-15 16:16:48.852, 11)  
(file8.tmp, e18f4c7fd255c350ce32222e2fb2e4bb, 2000-05-15 16:16:48.952, 12)  
(file10.tmp, c32563ac090d1c7c25b169312b74c0c4, 2000-05-15 16:21:51.808, 13)  
(file11.tmp, 4eb16eeaf5300e2ecfd4c3446508f35c, 2000-05-15 16:21:51.878, 14)
```

```
*** wretrieve SET INFORMATION ***  
(SET NAME, SET ID)  
(set78alias, 2)
```

*Retrieves all files in the set with the alias set78alias in the d:\archiveRetrieve directory. The output contains the set information and the minimal info for all files in the set. Since the -l option was not present the files are actually copied to the retrieve directory.*

E. D:\worm\test>D:\worm\bin\wretrieve -s set78alias -l  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\* wretrieve SET INFORMATION \*\*\*  
(SET NAME, SET ID)  
(set78alias, 2)

```
(NAME, WORM PATH)  
(file7.tmp,  
D:\arhive_home\pool3_underroot\pool3_path\b8028db8268065c6c59511326b5  
088bb)  
(file8.tmp, D:\arhive_home\pool3_underroot\pool3_path\e18f4c7fd255c350ce32222e2fb  
2e4bb)  
(file10.tmp, D:\arhive_home\default_path\c32563ac090d1c7c25b169312b74c0c4)  
(file11.tmp, D:\arhive_home\default_path\4eb16eeaf5300e2ecfd4c3446508f35c)
```

*Retrieves the archive paths for all files in the set with the alias set78alias. The output contains the set information and the WORM path for all files in the set. These paths together with the original file names can be used to create links to the archived files.*

## 6.2.4 wretrievec

Retrieves one or multiple files from the WORM system, the copy operation is performed by the client application as opposed to wretrieve, where the server copies the file to the client area.

```
wretrievec    -f file1 file2 file3 ...
              [-i fileId1, fileId2, fileId3 ...]
              [-s] set alias
              [-r] directory to retrieve the files into
              [-help]
```

### Description:

Client tool used to retrieve one or multiple files from the WORM archive. Can copy the files into the client indicated area. The need for server RW permissions to the retrieve directory is eliminated by using this client command tool. The client is executing the copy operation. Client needs Read permissions to the archive area.

If conflicts exist (multiple file names for a given digest or the same file name under 2 digests) the part of the request that is unambiguous will succeed and information to help solve the conflicts will be displayed. If multiple versions of one file exist, they need to be narrowed down to one version, by using the file internal id instead of its name. This information can be obtained by using the **wsearch** client tool.

A set of files that were archived as a set, can also be retrieved as a set, thus avoiding the need of repeating the wretrieve command for each member of the set.

The original file name will be used to copy the file out of the archive, since the server generated file name is meaningless for the user.

### Options:

- f** Files to be retrieved from the archive. One or more file names can be provided. Possible scenarios:
  - C. If only one file exists with a given name it will be retrieved in the current directory or the retrieve directory specified by the -r option. Since the client will perform the request, no server access is needed to the retrieve directory.
  - D. If multiple files with the same name exist in the WORM, they will have different digests, so a conflict will occur. The information that will uniquely identify each file in the conflict will be displayed. This scenario covers the situation in which multiple versions of the same file exist in the WORM.
- i** one or multiple Ids for the files to be retrieved from the archive. Specifying internal IDs is optional. A valid entity id can be obtained by using a wsearch operation based on file metadata (keywords, file name, owner, etc). The internal ID is a cleaner way to uniquely identify a file in the WORM. The actual uniqueness is guaranteed by the timestamp together with the file digest.
- t** timestamp for the file to be retrieved. Specifying a timestamp is optional. A valid timestamp can be obtained by using a wsearch operation based on file metadata (keywords, file name, owner, etc). The timestamp together with the file digest (specified with the -d option) uniquely identify a file in the WORM.
- d** digest for the file to be retrieved. Specifying a digest is optional. A valid digest can be obtained by using a wsearch operation based on file metadata (keywords, file name, owner, etc). Can be used individually or in conjunction with the -t flag. MD5 or SHA algorithms are used to compute the digest.
- p** Pool name where the file(s) were archived. If no pool name is provided, the default pool defined in the server configuration file will be first

used to search for the desired file(s). If that fails, a search based on the filename will be initiated. Using a pool name might speed up the catalog search operations.

**-s** Set id to be used for retrieval. It allows files that have been archived as one unit (a set) to be retrieved the same way - as opposed to issuing an individual command for each member of the set. It can be used on retrieval only if the same flag has been used on archival, when multiple files were archived through one operation. Tightly correlated files can be archived and retrieved with this option. Cannot be used together with the **-l** option, since the links have individual target names that would have to be specified.

**-a** Set alias to be used for retrieval. It allows files that have been archived as one unit (a set) to be retrieved the same way - as opposed to issuing an individual command for each member of the set. It can be used on retrieval only if an alias has been provided on archival, when multiple files were archived through one operation. Tightly correlated files can be archived and retrieved with this option. Cannot be used together with the **-l** option, since the links have individual target names that would have to be specified. If multiple sets have the same alias the command will fail and one unique set will have to be selected and retrieved using the set id flag.

**-r** Fully qualified output directory where the files will be retrieved. The wormserver daemon needs to have read/write permissions to that area. If no output directory is specified, the current client directory will be used.

#### Examples:

**A.** D:\worm\test>D:\worm\bin\wretrievec -f file1.tmp -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\* wretrieve RETRIEVE CONFLICTS \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)  
(file1.tmp, df223a55ce9e6c358204abdce7af31df, 2000-05-15 16:00:33.049, 5)

*Retrieves the file that was stored under the original name file1.tmp. In this scenario, the WORM archive contains two different digests for file1.tmp. That means that the same file has multiple versions that have been archived. The information that would help the user solve the conflict will be displayed. The client can retrieve the desired versions based on the internal id.*

**B.** D:\worm\test>D:\worm\bin\wretrievec -i 6 -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\* wretrievec RETRIEVED FILES \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)

*Retrieves the file having 6 as an internal id from the WORM archive. The output contains minimal information about the file that was retrieved.*

**C.** D:\worm\test>D:\worm\bin\wretrievec -s set78alias -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\* wretrieve RETRIEVED FILES \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file7.tmp, b8028db8268065c6c59511326b5088bb, 2000-05-15 16:16:48.852, 11)  
(file8.tmp, e18f4c7fd255c350ce32222e2fb2e4bb, 2000-05-15 16:16:48.952, 12)  
(file10.tmp, c32563ac090d1c7c25b169312b74c0c4, 2000-05-15 16:21:51.808, 13)

(file11.tmp, 4eb16eeaf5300e2ecfd4c3446508f35c, 2000-05-15 16:21:51.878, 14)

```
*** wretrieve SET INFORMATION ***  
(SET NAME, SET ID)  
(set78alias, 2)
```

Retrieves all files in the set with the alias set78alias in the d:\archiveRetrieve directory.  
The output contains the set information and the minimal info for all files in the set.

## 6.2.5 wlist

Generic purpose administration tool that displays information about entities in the archive. It can list information about pools in the archive, about files in a specific pool etc.

```
wlist [-p] poolname  
      [-s]  
      [-help]
```

### Description:

Client tool used to list all pools managed by a given worm server. Contents of a specific pool can be displayed. This tools list pool contents based on the original file name that was specified by the user at insertion time.

### Options:

If no options are specified, all pool names managed by the wormserver instances the client connects to are listed.

**-p**

**-s** One or multiple Ids for the files to be retrieved from the archive. Specifying internal IDs is optional. A valid entity id can be obtained by using a wsearch operation based on file metadata (keywords, file name, owner, etc). The internal ID is a cleaner way to uniquely identify a file in the WORM. The actual uniqueness is guaranteed by the timestamp together with the file digest.

**-help** displays the usage information.

### Examples:

**A.** D:\worm\test>D:\worm\bin\wretrievec -f file1.tmp -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\* wretrieve RETRIEVE CONFLICTS \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)  
(file1.tmp, df223a55ce9e6c358204abdce7af31df, 2000-05-15 16:00:33.049, 5)

*Retrieves the file that was stored under the original name file1.tmp. In this scenario, the WORM archive contains two different digests for file1.tmp. That means that the same file has multiple versions that have been archived. The information that would help the user solve the conflict will be displayed. The client can retrieve the desired versions based on the internal id.*

**B.** D:\worm\test>D:\worm\bin\wretrievec -i 6 -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local  
\*\*\* wretrievec RETRIEVED FILES \*\*\*  
(NAME, DIGEST, INSERT DATE, ID)  
(file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)

Retrieves the file having 6 as an internal id from the WORM archive. The output contains minimal information about the file that was retrieved. Since the `-l` option was not present the file was actually copied to the retrieve directory.

## 6.2.6 wdelete

Deletes ND-WORM entities, including pools, sets or files. All references to the specified entities will be removed from the ND-WORM system.

**wdelete**      **[-f] file1 file2 file3 ...**  
                 **[-p] poolname1 poolname2 poolname3 ...**  
                 **[-i] fileId1 fileId2, fileId3 ...**  
                 **[-s] setalias1 setalias2 setalias3 ...**  
                 **[-o]**  
                 **[-help]**

### Description:

Client tool used to remove different types of entities from the ND-WORM system. Should be used with care since the deletion is an irreversible process. Individual files, file sets and entire pools can be deleted. If files that reside in a pool have references from other pools, no deletes will be performed. In these situations, the references will have to be removed manually before the actual files are removed.

*Important Note: No file is removed from a given pool unless all references to the file are from the same pool. The `-o` flag is used to overwrite this behavior and remove all references to the file being deleted. If a file is part of a set, the set will be also removed if the file being removed is the only set member.*

### Options:

- f**      files to be removed from the archive. If references to those file exist from other pools (because of the Non Duplicate feature of this worm), the remove operation will fail. This is the default behavior, that can be overwritten by using the `-o` flag.
- p**      pool names for the pools that need to be removed from the archive. By removing a pool, all entities that belong to that pool (files, sets) will also be removed from the ND-WORM system.
- s**      one or multiple Ids for the files to be removed from the archive. If references to those file exist from other pools (because of the Non Duplicate feature of this worm), the remove operation will fail. This is the default behavior, that can be overwritten by using the `-o` flag.
- o**      overwrite flag. Used to overwrite the default command behavior, which is to fail the remove operation if references to the entities being removed exist from other pool. By specifying the `-o` flag, deleting an entity will also delete all references to the entity, regardless if they come from the same pool or another pool.
- help**    displays the help dialog.

## 6.2.7 wsync

Maintenance tool that performs a synchronization between the database catalog entries and the actual entities that exist on disk. Since it is not possible to prevent direct disk access (without using the ND-WORM client tools) the potential of the database and the disk being out of sync exists.

**wsync**    **[-r]**  
                 **-help**

**Description:**

Check all the database entries against actual files being stored on disk and prints out a report. All entries that have no disk correspondence will be removed from the database. The sync operation should be performed periodically. If there are entries on disk that are not in the ND-WORM catalog, they will be reported in the command output.

**Options:**

If no options are present, the tool does not remove any database entries, it just prints out a report.

**-r** removes database entries that have to corresponding disk entities. This is the flag to be used when the actual sync operation is to be performed.

**-help** Displays the help dialog.

**Examples:**

A. D:\worm\test>D:\worm\bin\wretrievec -f file1.tmp -r d:\archiveRetrieve

*Retrieves the file that was stored under the original name file1.tmp. In this scenario, the WORM archive contains two different digests for file1.tmp. That means that the same file has multiple versions that have been archived. The information that would help the user solve the conflict will be displayed. The client can retrieve the desired versions based on the internal id.*

B. D:\worm\test>D:\worm\bin\wretrievec -i 6 -r d:\archiveRetrieve  
Configuration file: d:\worm\bin\worm.config.client.local

```
*** wretrievec RETRIEVED FILES ***
(NAME, DIGEST, INSERT DATE, ID)
(file1.tmp, 33e8def83e25c9f2dad53db52783d8e3, 2000-05-15 16:01:19.216, 6)
```

*Retrieves the file having 6 as an internal id from the WORM archive. The output contains minimal information about the file that was retrieved. Since the -l option was not present the file was actually copied to the retrieve directory.*

## 7 Changes since last Release

```
*****
**** WORM 1.0 May 31st 2000 *****
*****
```

Basic proptotype functionality: wormserver, winsert, wsearch, wretrieve

```
*****
**** WORM 1.1 June 30th 2000 *****
*****
```

added wretrieveC - client copies file out of the archive, rather than server. Only read permissions are needed for the client account in the WORM archive area.  
Works with -f, -i, -s and -r options.

more robust logging - until now, all client tools would log information in the same file. Now there is a file in the worm.jar called Resources.properties in the jar file, that contains client tool log prefixes. The following postfixes are added to the MESSAGELOG entry in the client configuration file:  
WI - for winsert

WS - for search  
WR - for retrieve  
WRC - for wretrieve

dynamic versioning - the current code version is now contained in the jar file

winsert enhancement - included file name in the output

fixed following bugs - correct help message in wretrieve and wretrieveC, if  
-s option is used, a set ALIAS not a set ID is required.  
All files in the set with the specified alias will be  
retrieved.  
- cleaned up the WormCommonConstants source file

```
*****  
**** WORM 1.2 ****  
*****
```

wlist - new tool to list pools and their contents

wdelete – removes pool entities

wsync -

user manual - added more documentation to the user manual to reflect changes