HYPERtape OpenMedia *System Manager Guide*

Product number **M7120**
Version 3.1 Rev. 1 **March 1998**

MultiStream Systems
The edge in network data management

M7120 HYPERtape OpenMedia **System Manager Guide**

Revision history

Revision	Date	Software release
3.0 Rev. 3	Oct 1997	3.0 level 10
3.1 Rev. 1	Mar 1998	3.1 level 10

Contacts

Europe

MultiStream Systems GmbH
Junkerstrasse 1
D-82178 Puchheim, Germany

Tel: +49 89 890 162 0
Fax: +49 89 890 162 62
Email multistream@compuserve.com
Internet: http://www.multistream.com

North America

MultiStream Systems, Inc.
4640 West 77th Street, Suite 100
Edina, MN 55435 USA

Tel: +1 612 835 2475
Tel: +1 612 835 2537
Email multistream@compuserve.com
Internet: http://www.multistream.com

Questions and comments

Please contact your local MultiStream support office or your software supplier if you require help or advice on the product or have any comments on this manual.

Copyright information

© Copyright 1998 MultiStream Systems, Inc.

All rights reserved.

This document refers to numerous products by their trade names. In most, if not in all cases, these designations are claimed as trademarks or registered trademarks by their respective customers. The references are made for informational purposes only; no consent is implied.

HYPERtape and OpenMedia are registered trademarks of MultiStream Systems, Inc.

About this manual

This manual is intended to help system managers and administrators install HYPERtape OpenMedia, and monitor and control the storage and backup systems of the enterprise using the Graphical User Interface (GUI) and Command Line Interface (CLI).

Structure

The manual has two parts: the first part introduces HYPERtape OpenMedia and explains how to install it and use the GUI and CLI. The second part discusses storage management and administration policies and shows how to implement such policies by manipulating the database.

The manual has the following chapters and appendices:

Chapter 1	**Introducing HYPERtape OpenMedia** Describes OpenMedia, lists its main features and describes the uses of the GUI and CLI.
Chapter 2	**Setting up HYPERtape OpenMedia** Explains how to install and configure OpenMedia to operate with jukeboxes, tapes and other HYPERtape components.
Chapter 3	**Using the GUI** Describes the GUI and explains how to perform basic operations such as drag and drop. This chapter also explains how to add, modify and delete objects in the OpenMedia database.
Chapter 4	**Using the CLI** Describes the CLI and its uses, explains the syntax, shows how to access online help, categorizes and lists the commands available and provides examples of typical CLI scripts.
Chapter 5	**Developing a storage management strategy** Explains how to design a storage and backup management strategy.
Chapter 6	**Seting up and managing the object database** Explains how to set up and organize administration objects so that you can implement your storage management policy and monitor its progress.
Chapter 7	**Managing media operations** Explains how to set up and manage jukeboxes, media locations and devices.
Chapter 8	**Managing stored data** Explains how to use OpenMedia to manage the tapes, segments and savesets recorded in the object database.
Appendix A	Describes the function of the OpenMedia server.
Appendix B	Lists the messages output by OpenMedia.

Further reading

- *HYPERtape OpenMedia Concepts and Facilities*

This manual provides an overview of HYPERtape OpenMedia, explaining what it is and what it does. This manual is intended for anyone who needs an introduction to the product, from technical decision makers and system managers who are assessing HYPERtape OpenMedia, to software engineers who are developing applications that will integrate with it.

- *HYPERtape OpenMedia Application Program Interface*

This manual is a reference manual for application engineers. The HYPERtape OpenMedia Application Program Interface is largely self documenting via the header files of the various software components, so this manual concentrates on concepts, techniques and examples.

Need help?

HYPERtape OpenMedia is designed to be easy to use and operate. However, if you have questions about the product or encounter a problem with the software, please contact your supplier or the customer support office nearest you for assistance.

Before reporting a software problem, record the commands that preceded the error and the exact error message that appeared. When you have this information on hand, the support staff are better able to help you with the problem.

Conventions

To help you locate and interpret information easily, this manual uses consistent visual cues and a few standard text formats. These conventions are explained below.

Example	Usage
`Linking...`	Computer output and command formats are shown using this monospaced typeface.
`SHOW`	In examples, command names and items to be entered verbatim use the monospaced typeface in bold. It appears with regular monospaced typeface to illustrate computer output and the responses you enter.
[JUKEBOX]	An item enclosed in square brackets is an optional (not a required) entry.
object_type	If a value in an example is shown in italics, replace it with the actual parameter you use.
{YES | NO}	When values are shown in curly brackets, separated by a vertical line, you should choose and enter one of the items.
RETURN	In text, the name of a key that you press appears in uppercase letters.
<ESC>	In examples, angle brackets enclose the names of keys that you press
Ctrl+X	A key combination, shown in uppercase with a slash separating two key names, indicates that you hold down the first key while pressing the second key. For example, Ctrl+X indicates that you hold down Ctrl while you press X.
HYPERtape User Manual	Cross references, including those to other documents, are shown in italics.

Contents

About this manual **3**
Structure 3
Further reading 4
Need help? 4
Conventions 4
1 Introducing HYPERtape OpenMedia **11**
What is HYPERtape OpenMedia? 11
Object database 12
System architecture 12
Graphical User Interface (GUI) 13
Command Line Interface (CLI) 13
Application Program Interface (API) 14
2 Setting up HYPERtape OpenMedia **15**
Installing HYPERtape OpenMedia 15
Populating the database 15
Local configuration file 20
Starting and stopping the OpenMedia service 21
Before starting the OpenMedia service for the first time 21
Starting the OpenMedia service on Windows NT hosts 21
Starting the OpenMedia service on Windows 95 hosts 22
Starting the OpenMedia service on UNIX hosts 22
Starting the OpenMedia service on OpenVMS hosts 22
Configuring HYPERtape OpenMedia 23
Configuring the database 23
Integrating with robot managers 25
Integrating with SCH:Robot 26
Initializing tapes for use with HYPERtape 27
Setting up a HYPERtape backup object 28
Setting up a HYPERtape Backup Node 31
Removing HYPERtape OpenMedia 31
Windows NT (Intel) hosts 31
Windows NT (Alpha) hosts 31
Windows 95 hosts 31
UNIX hosts 32
OpenVMS hosts 32

3 Using the GUI 33
Starting the GUI 33
Main window 34
Menu items 35
Logging in 35
Finding your way round the GUI 36
Resizing boxes 38
Reducing windows to icons 39
Using wildcards in selection windows 39
Using browsers 39
Entering data into detail windows 41
Standard command buttons 44
Notification boxes 45
Getting help 45
Managing database objects 46
Listing database objects 46
Adding database objects 46
Adding related objects 47
Modifying database objects 48
Deleting database objects 48
Printing reports 49
Setting preferences 50
Leaving the GUI 50

4 Using the CLI 51
Introducing the CLI 51
CLI architecture 51
CLI command categories 51
Commands that interrogate or manipulate database objects 51
Commands that manage media objects 52
Commands that process prompts and changes 52
Commands that make server requests 53
Miscellaneous commands 53
Starting and stopping the CLI 53
General syntax 54
Getting help 55
Using the DISPLAY command 55
Using abbreviations 56
Selection criteria 56
Embedded selection criteria 57
Tagged selection criteria 58
Data types 58
Item lists 61
Setting defaults 61
CLI commands 62
Using ASSIGN and DEASSIGN commands 82
Writing scripts 82
Using SHOW and SET commands 83

5 Developing a storage management strategy 85
Storage management requirements 85
Device and media management 86
Retention of data 86
Storing tapes offsite 86
Developing a backup strategy 87
Limitations affecting a backup strategy 88
Types of backup 89
Frequency of backups 90
Backup retention 90
Offsite storage 91
Implementing your storage management strategy 91

6 Setting up and managing the object database 93
Initial considerations 93
Organizing the OpenMedia database 93
General administration 94
Using media pools to organize tapes 95
Manipulating locality objects 96
Locality window 96
Viewing brokers for the locality 97
Manipulating host objects 97
Host window 98
Viewing devices on the host 99
Viewing localities brokered by the host 99
Manipulating physical type objects 99
Physical type window 100
Manipulating media type objects 100
Media type window 101
Viewing devices that read the media type 101
Viewing devices that write the media type 102
Manipulating media pool objects 102
Media pool window 103
Setting system policies 104
System policy window 105
User administration 106
Manipulating user objects 106
User window 106
Manipulating user account objects 107
User account window 108
Manipulating privilege objects 108
Privilege window 109
Recovery from lost users and privileges 109

7 Managing media operations 111
Manipulating database media objects 111
Manipulating jukebox objects 111
Jukebox window 112
Manipulating media location objects 114
Media location window 114

Viewing device preferences 115
Manipulating device objects 115
Device window 116
Device statistics 117
Viewing media types read by the device 118
Viewing media types written by the device 118
Viewing device preferences by location 119
Viewing hosts that access the device 119
Backing up the OpenMedia database 120

8 Managing stored data 121

Tape and saveset management capabilities 121
Managing the movement of tapes 122
Identifying the tapes to be moved 122
Updating the onsite media locations 122
Onsite and offsite due dates 122
Scratching tapes 123
Tracking tapes 123
Daily processing 123
Using media operations 124
Scratching expired tapes 124
Expiring savesets 125
Removing expired savesets 125
Marking tapes offsite 126
Marking tapes onsite 126
Marking devices cleaned 127
Marking devices serviced 127
Reviewing prompts 127
Manipulating tapes, segments and savesets 129
Manipulating tape objects 129
Tape window 130
Tape location information 132
Tape usage information 133
Manipulating segment objects 134
Segment window 134
Manipulating saveset objects 135
Saveset window 135
Interrogations 136
Using the Interrogations options 136
Interrogation options 137
Monitoring media activities 138
Looking up reserved devices 138
Looking up reserved tapes 138
Looking up opened savesets 139
Looking up device statuses 139
Looking up tape statuses 139
Looking up connections 139
Running site defined reports 140

A HYPERtape OpenMedia server 141
Operation of the server 141
Startup and shutdown 141
Security and privilege 142
Error and disaster recovery 142
Lost connection to a client 142
Abnormal termination of the server 142
B HYPERtape OpenMedia messages 143
Message format 143
Message list 143
Glossary 167
Index 175

1

Introducing HYPERtape OpenMedia

What is HYPERtape OpenMedia?

HYPERtape is a multi-vendor, multi-protocol software solution for automated, unattended network backup, archiving and data recovery throughout computing enterprises. It protects your network investment by using your existing hardware and software components and adapting easily to organizational and operational changes.

Since it supports an unlimited number of concurrent sessions between any number of network nodes, this solution is ideal for high-speed network backup. HYPERtape backs up workstations, departmental servers and enterprise servers across entire multi-platform networks. It offers an unlimited choice of backup methods through its support of virtually all storage media.

OpenMedia is a key component of MultiStream's new generation of storage management solutions, which aim to provide the best enterprise-wide open systems storage management infrastructure in the market place. It is a software tool for managing storage media across large and small enterprises that fulfills customers' needs for scaleable and flexible solutions in the areas of multiplatform network backup, network attached data storage, and multimedia Intranet and Internet communications.

It is fully year 2000 compliant.

The main facilities offered by OpenMedia are:

- Volume management
- Saveset management
- Device management
- Robotics or human operator control
- Media user and host authorizations
- System policy application

Object database

OpenMedia maintains a database of *media objects* and *administration objects*, which can be physical items such as the volumes and host systems in your enterprise, or abstract entities such as media pools and user privileges. Objects and the defined relationships between them are encompassed in the OpenMedia *model*.

OpenMedia provides you with the tools you need to describe your own organization and to monitor, control and automate your storage management. For more information about the OpenMedia model, see the manual *HYPERtape OpenMedia Concepts and Facilities*.

System architecture

OpenMedia software is implemented using a client/server architecture and consists of the following components:

- *Server*
  Executes requests issued by clients and returns one or more responses.
- *Service broker*
  Connects clients with service providers
- *Jukebox controller*
  Services load, unload and eject requests to jukebox devices
- *Agent*
  Starts up the server, service broker and jukebox controller software on a host as necessary
- *Graphical User Interface (GUI)*
  A client application for manipulating objects
- *Command Line Interface (CLI)*
  A client application for manipulating objects via command scripts

The GUI and CLI have been developed using the OpenMedia *Application Program Interface* (API), which is also available for developing client applications. The GUI provides a higher level interface, while the CLI is closer to the low level objects and the internal API.

The server does most of the work of OpenMedia, in response to requests generated by clients. Only one server runs in the OpenMedia domain.

OpenMedia uses *dynamic* service broker lookup, whereby clients ask for a service rather than a particular server and are connected to any server offering that service. This means that only the service brokers need to know where items such as the server and jukebox controllers reside, so individual clients are not affected if such items are moved to new hosts. You can have a number of service brokers in the network, to provide resilience in the event of a failure and to avoid any client reconfiguration.

A jukebox controller runs on the host that controls the jukebox. There must be one jukebox controller for each jukebox controlled by the host.

The Agent runs on every host. It uses a locally held configuration file to determine whether the host should run service broker, server or jukebox controller software and starts up the required components accordingly.

Graphical User Interface (GUI)

The GUI is a client application that allows manipulation of objects in the database from a host workstation. For example, you can interrogate the database to discover which tapes are past their expiration dates, then scratch them ready for reuse.

The purpose of the GUI is to allow the System Manager to manage, interrogate and change data in the OpenMedia database. It provides facilities for:

- Interrogating the database
- Configuring the model
- Onsite/offsite operation
- Responding to prompts

The GUI can accept look-and-feel instructions to behave as Motif, OPENLOOK or Windows.

See Chapter 3 for information about using the GUI.

Command Line Interface (CLI)

The CLI is a client application that allows you to issue server requests directly from the command line. Its main use is for developing scripts for managing media and/or integrating OpenMedia with other applications. It can also be useful for remote support operations over dial up modems using connections such as telnet.

The CLI is a set of commands with associated requests, such as to reserve a tape. Supported operations include:

- Looking up, updating, adding and deleting database objects
- Creating and running reports
- Looking up volatile objects
- Shutting down the server
- Loading, reserving, releasing, unloading and looking up the status of tapes and devices
- Opening and closing savesets
- Opening and closing segments

See Chapter 4 for information about using the CLI.

Application Program Interface (API)

The API is a development tool that is designed to help Value Added Developers (VADs) and Original Equipment Manufacturers (OEMs) integrate OpenMedia with their own products. For example, a client application developed using the API might identify and allocate tapes before issuing load commands. The API is used internally by the GUI, CLI, Agent and jukebox controller components of OpenMedia. You do not need to purchase the API to use these components: this is necessary only if you want to write your own client applications. For more information about the API, see the manual *HYPERtape OpenMedia Application Program Interface*.

2

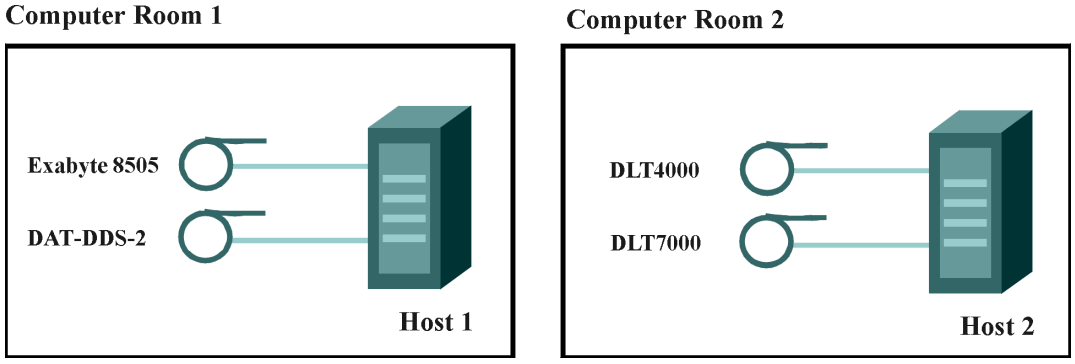
Setting up HYPERtape OpenMedia

Installing HYPERtape OpenMedia

OpenMedia is supplied in kit form on CD ROM. Included with the kit is a readme.txt file, which contains instructions for installing and upgrading OpenMedia. Follow these instructions to install OpenMedia. In particular, note the privileges you will need to perform the installation and the information you will be required to supply during the installation procedure.

Populating the database

During the initial installation of a server host, the OpenMedia database is automatically created. It is populated with the reports used by the GUI. No tape, segment or saveset objects are created, but samples of several other objects, based on two example computer rooms, are created to provide you with an example of how OpenMedia models real systems.



In addition, the installation procedure creates sets of standard physical types, media types and system policies.

The following sections describe the objects created by the installation procedure:

Localities

The single locality created is given the name entered during installation.

Users

A user object is created with the name *manager* and the password *secret*. For security reasons, you should start up the GUI or CLI and change this password immediately after installation.

User accounts

User account objects are created for the *manager* user with native users *Administrator* and *SYSTEM* on the server host.

Privileges

A privilege object is created for the *manager* user, with *MANAGER* and *OPERATOR* privileges.

Hosts

A host object is created for the host on which the installation procedure runs. In addition, a number of sample host objects are created. You can find descriptions of these objects in the file add-hosts.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. The following are examples from the add-hosts.txt file; the name of each host is followed by the settings for the unique name, the flag that indicates whether the host runs server software and comms protocol1:

Example Host 1
unique name: Example Host 1
runs admed server: NO
comms protocol1: tcp

Example Host 2
unique name: Example Host 2
runs admed server: NO
comms protocol1: tcp

Physical types

A set of standard physical type objects are created. You can find descriptions of these objects in the file add-physicaltypes.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. If you need a type of tape that is not in this list, please contact your supplier or local support office for advice.

Media types

A set of standard media type objects are created. You can find descriptions of these objects in the file add-mediatypes.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host.

Jukeboxes

Two sample jukebox objects are created, representing two manual operator-controlled computer rooms. You can find descriptions of these objects in the file add-jukeboxes.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. The following are examples from the add-jukeboxes.txt file; the name of each object is followed by the settings for jukebox class (*OPERATOR* or *GENERIC*) and locked indicator. The examples are locked to avoid accidental use.

Example Computer Room 1 Drives
class: OPERATOR
locked: YES

Example Computer Room 2 Drives
class: OPERATOR
locked: YES

Media locations

Three sample media location objects are created. You can find descriptions of these objects in the file add-medialocations.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. The following are examples from the add-medialocations.txt file; the name of each object is followed by the name of the jukebox, if any, with which the media location is associated:

Example Computer Room 1
jukebox: Example Computer Room 1 Drives

Example Computer Room 2
jukebox: Example Computer Room 2 Drives

Offsite
(no jukebox set)

Media pools

Three sample media pool objects are created. You can find descriptions of these objects in the file add-mediapools.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. The following are examples from the add-mediapools.txt file; the media pool name is followed by the values (if any) set for the media pool type, associated scratch pool, default onsite and offsite media locations, retention, onsite duration, offsite duration and movement cycle:

Scratch
type: SCRATCH
retention: 0
onsite duration: 0

Example Public 1
type: PUBLIC
associated scratch pool: Scratch
retention: +360
default onsite media location: Example Computer Room 1
onsite duration: 0
default offsite media location: Offsite
offsite duration: 0
movement cycle: BY_USAGE

Example Public 2
type: PUBLIC
associated scratch pool: Scratch
retention: +360
default onsite media location: Example Computer Room 2
onsite duration: 0
default offsite media location: Offsite
offsite duration: 0
movement cycle: BY_USAGE

Devices

A number of sample device objects are created. You can find descriptions of these objects in the file add-devices.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. The following are examples from the add-devices.txt file; the name of each object is followed by the settings for the physical name, jukebox name and locked indicator:

Example Exabyte 8505
physical name: Tape0
jukebox: Example Computer Room 1 Drives
locked: YES

Example 4mm DAT DDS-2
physical name: Tape1
jukebox: Example Computer Room 1 Drives
locked: YES

Example DLT 7000
physical name: Tape7
jukebox: Example Computer Room 2 Drives
locked: YES

Example DLT 4000
physical name: Tape9
jukebox: Example Computer Room 2 Drives
locked: YES

Media types associated with devices

A number of sample objects are created that detail the media types that can be read and written by each device created during installation. You can find descriptions of these objects in the file add-devices.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. For example, the following devices can read and write the following media types:

Device	Read media types	Write media types
Example Exabyte 8505	8MM-600MB	8MM-600MB
	8MM-1GB-C	8MM-1GB-C
	8MM-2GB	8MM-2GB
	8MM-4GB-C	8MM-4GB-C
	8MM-5GB	8MM-5GB
	8MM-10GB-C	8MM-10GB-C
Example 4mm DAT DDS-2	4MM-1GB	
	4MM-2GB-C	
	4MM-2GB	
	4MM-4GB-C	
	4MM-DDS2-4GB	4MM-DDS2-4GB
	4MM-DDS2-8GB-C	4MM-DDS2-8GB-C

Device	Read media types	Write media types
Example DLT 4000	DLT2000-10GB	DLT2000-10GB
	DLT2000-20GB-C	DLT2000-20GB-C
	DLT2000XT-15GB	DLT2000XT-15GB
	DLT2000XT-30GB-C	DLT2000XT-30GB-C
	DLT4000-20GB	DLT4000-20GB
	DLT4000-40GB-C	DLT4000-40GB-C
Example DLT 7000	DLT2000-10GB	DLT2000-10GB
	DLT2000-20GB-C	DLT2000-20GB-C
	DLT2000XT-15GB	DLT2000XT-15GB
	DLT2000XT-30GB-C	DLT2000XT-30GB-C
	DLT4000-20GB	DLT4000-20GB
	DLT4000-40GB-C	DLT4000-40GB-C
	DLT7000-35GB	DLT7000-35GB
	DLT7000-70GB-C	DLT7000-70GB-C

Devices accessible by hosts

A number of sample objects are created that detail the hosts that can access each device created during installation. You can find descriptions of these objects in the file add-devices.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. For example, the following devices can be accessed by the following hosts:

device: Example Exabyte 8505
    host: Example Host 1
device: Example 4mm DAT DDS-2
    host: Example Host 1
device: Example DLT 7000
    host: Example Host 2
device: Example DLT 4000
    host: Example Host 2

System policies

A set of standard system policy objects are created. You can find descriptions of these objects in the file add-systempolicies.txt, which is added to the bin subdirectory of the OpenMedia top level directory when you install OpenMedia on a server host. The following are examples from the add-systempolicies.txt file; the name of each object is followed by a brief description and the default setting of the system policy:

DB_MAX_HANDLERS
    Maximum database statement handlers
    default: 4
JBX_MAX_HANDLERS
    Maximum jukebox script processes
    default: 5
JOB_DAILY_PROCESSING
    Time, in the format *hh*:*mm* at which daily processing should run
    Default: 05:00

LM_LOADTAPE_TIMEOUT
Load tape timeout in seconds
default: 3600
MM_DEFAULT_CONFIRM_DELAY
Maximum wait time for a tape to come online, in seconds
default: 10
MM_DEFAULT_CONFIRM_RETRIES
Maximum number of checks that a tape is online
default: 15
MM_DEVICE_ALLOCATION_METHOD
Device allocation method
default: LeastOftenUsed
MM_MAX_LOAD_TRIES
Maximum number of attempts to load a tape
default: 5
NET_RECON_TIMEOUT
Network reconnection timeout in seconds
default: 120
TRACE_ERROR_HISTORY_KB
Size of the trace buffer in kilobytes
default: 30
TRACE_MAX_FILE_KB
Maximum trace log file size in kilobytes
default: 500

Local configuration file

The installation procedure creates a local configuration file called LOCAL_CONFIG.DAT in the data subdirectory of the OpenMedia home directory. This file contains information required to start OpenMedia on the host, and to enable client applications such as the GUI or CLI to connect to the service broker. It defines values for the following keywords:

- LOCALHOOD
  The name of the neighborhood locality to which the host belongs.
- LOCALADMIN
  Set to Yes if the host runs the server software, No otherwise.
- PROTOCOL
  The network protocol (tcp).
- LOCALBROKER
  The name of a host running service broker software.

You can have more than one LOCALBROKER defined.

A host that runs any combination server, jukebox controller and service broker software is said to run an *OpenMedia service*. When an OpenMedia service starts, the local configuration file is updated automatically from the database. It should never be necessary to edit the file; indeed, any edits that you make will be lost when the service is next started. Hosts that run an OpenMedia service or HYPERtape Backup Nodes must have host objects defined in the OpenMedia database.

Hosts that only run a GUI and/or CLI do not require a host object defined in the database, however you must define the host object if you want the local configuration file to be automatically updated. Otherwise, you must use a text editor to maintain the file.

Starting and stopping the OpenMedia service

Before starting the OpenMedia service for the first time

Jukebox controllers and HYPERtape Backup Nodes rely on automatic accreditation via host and user account objects. The installation procedure automatically adds these objects to the OpenMedia database for the server host, but you must add them yourself for client hosts before you start the OpenMedia service or run a HYPERtape Backup Node for the first time on the host.

Use the GUI or CLI to log in to as the *manager* user to the locality specified during server installation, then create a host object for the host and a user account object that allows the host to login automatically. Use the **Help/On version** option in the GUI or the CLI SHOW command with the ENVIRONMENT option to check the full name of the host, which is required for the unique name field of the host object and the host field in the user account object.

The native user in the user account object must identify the user under which the software is started. This is *SYSTEM* for Windows NT and OpenVMS systems and *root* for UNIX systems, . For more information about manipulating these objects, see the sections *Manipulating host objects* and *Manipulating user account objects* in Chapter 6.

Every time the OpenMedia service starts, the host's local configuration file is updated to reflect the current state of the relevant database objects.

While it is not essential to create a host and user account object for hosts that will only run the GUI or CLI, failing to create a host object prevents the automatic update of the host's local configuration file and causes error messages if the OpenMedia service is started on the host.

Starting the OpenMedia service on Windows NT hosts

The OpenMedia service is usually started automatically at system startup. To start or stop the OpenMedia service manually:

1 Launch the Services applet in Control Panel

2 Select the HYPERtape OpenMedia Manager service

3 Click on the Start or Stop button as appropriate.

When you start or stop the OpenMedia service, the following software is started or stopped, as required for the host:

- Service broker
- Server (Windows NT Intel only)
- Database server (Windows NT Intel only)
- Jukebox controller(s)

Whenever the OpenMedia service starts, the local configuration file is updated to reflect the state of the database objects.

Starting the OpenMedia service on Windows 95 hosts

1 Open an MS-DOS prompt window and set the default directory to the OpenMedia \bin directory by entering:

```
cd %ommhome%\bin
```

2 Start the OpenMedia service by entering:

```
admedagent -start
```

3 Stop the OpenMedia service by entering:

```
admedagent -stop
```

This starts or stops any service broker software required for the host and updates the local configuration file. You do not require these steps to start the CLI or GUI: simply click on the CLI or GUI shortcut in the HYPERtape OpenMedia directory, which is set up in the Programs directory of the Start menu at installation.

To automate OpenMedia service startup, edit the host's AUTOEXEC.BAT file to execute admedagent -start whenever Windows 95 starts.

Starting the OpenMedia service on UNIX hosts

1 Set the environment variables and path changes required by the OpenMedia service by entering:

```
. /etc/ommsetup.sh
```

for Bourne/Korn shells, or

```
source /etc/ommsetup.csh
```

for C shells.

The dot and source ensure that the environment variables are set for the current process.

3 Start the OpenMedia service by entering:

```
admedagent -start
```

4 Stop the OpenMedia service by entering:

```
admedagent -stop
```

This starts or stops any jukebox controllers and service broker software required for the host and updates the local configuration file.

To automate OpenMedia service startup, edit the host's startup scripts as appropriate to the particular UNIX implementation to execute admedagent -startup whenever UNIX starts.

Starting the OpenMedia service on OpenVMS hosts

1 Set the symbols required for the OpenMedia service by executing SYS$MANAGER:OMM_LOGIN.COM. Enter:

```
@SYS$MANAGER:OMM_LOGIN
```

2 Start the OpenMedia service by entering:

```
admedagent /start
```

3 Stop the OpenMedia service by entering:

```
admedagent /stop
```

This starts or stops any jukebox controllers and service broker software required for the host and updates the local configuration file.

To automate the procedure, edit the host's system startup command procedure to execute SYS$MANAGER:OMM_LOGIN.COM and admedagent /start whenever OpenVMS starts. Edit the host's shutdown command procedure to execute SYS$MANAGER:OMM_LOGIN.COM and admedagent /stop whenever OpenVMS shuts down.

Configuring HYPERtape OpenMedia

Once you have installed OpenMedia for the first time, you need to set up the OpenMedia database to describe the items in your enterprise and the relationships between them. This section explains how to configure the database for your jukeboxes, tape drives and tapes, and how to integrate OpenMedia with your robot manager and backup software, in particular the HYPERtape Robot Manager (HTRM) and HYPERtape Backup Nodes. If you are not familiar with the OpenMedia model, see the manual *HYPERtape OpenMedia Concepts and Facilities*.

Configuring the database

You can add objects to the OpenMedia database using the GUI or the CLI commands ADD and COPY. For a general explanation about how to add database objects, see *Adding database objects* in Chapter 3. As well as explaining how to add database objects via the GUI, the section lists the dependencies of each object type so that you can plan the order in which you introduce new objects to the database. For example, a device object requires a reference to a jukebox, so you should add the jukebox object before you try to add objects for the devices it contains. For information about the CLI commands, see *ADD* and *COPY* in Chapter 4.

The OpenMedia database already contains a number of sample objects, which are added at installation (see *Populating the database* earlier in this chapter for details). The easiest way to add a database object is to use the GUI to find a sample object that is similar to the one you require, amend it as necessary and then add the amended object to the database by clicking on the **Add** button (see *Adding database objects* in Chapter 3 for details).

When you add objects that describe robotic jukeboxes, there are other factors that you must take into account in addition to the mandatory references described in *Adding database objects*. In particular, you must set the control host that runs the robot manager software for the jukebox, and the media location to which tapes default when they are removed from the jukebox.

Proceed as follows to configure OpenMedia for use with both robotic and operator controlled jukeboxes, adding objects in the order specified:

1 Add host objects, if they are not already defined, for robotic jukebox control hosts, HYPERtape Backup Nodes and any other hosts that are to access drives. See *Manipulating host objects* in Chapter 6 for detailed information about host objects.

The unique name field in the host object must match the name by which the network knows the host. You can check on this via the GUI by selecting the **On version** option on the **Help** menu to display the Welcome window. This shows both the host's native username and the network name of the host. Alternatively, use the CLI command SHOW with the ENVIRONMENT option, as described in Chapter 4.

2 Add user account objects, if they are not already defined, to allow jukebox controller and Backup Node hosts to log in automatically. Specify the host and the native username under which the software is started. This is usually *SYSTEM* for Windows NT and OpenVMS systems and *root* for UNIX systems. For information about these objects, see the sections *Manipulating host objects* and *Manipulating user account objects* in Chapter 6.

3 Set up the jukebox objects. For a robotic jukebox, set the class as *GENERIC* and the model field to *htrm* for HYPERtape Robot Manager, *sch* for SCH:Robot, *ACSLS* for ACSLS or oasis for Oasis RLM. For an operator controlled jukebox such as a computer room, leave the model blank and specify the class as OPERATOR. See *Manipulating jukebox objects* in Chapter 7 for detailed information about jukebox objects.

   If you use the OpenMedia's sample objects as the basis for your jukebox objects, remember to clear the locked flag before adding them to the database.

4 Set up media location objects that specify the dropped media location for any robotic jukeboxes. If OpenMedia cannot find a tape that it expects to be in the jukebox, it defaults the tape's current media location to this dropped media location, which should describe an area outside the jukebox. See *Manipulating media location objects* in Chapter 7 for detailed information about media location objects.

5 Set up media location objects for each jukebox. These are the media locations given to tapes when they are inside the jukebox. See *Manipulating media location objects* in Chapter 7 for detailed information about media location objects.

6 Add device objects for each jukebox drive. OpenMedia's sample device objects describe many common drives: use the GUI to find the nearest match to your requirements, then repeatedly modify and add the objects you require.

   When you copy devices using the GUI, any related objects that specify the readable and writeable media types and the names of any hosts that can access the device are automatically duplicated. Modify these objects for each device as required.

7 Add tape objects to the database. This is easiest performed through a jukebox manager and its reconciliation program or by a conversion procedure from a previous product.

   The method used for jukebox reconciliation depends on the robot manager. If you are using HYPERtape Robot Manager or the HYPERtape Robotic Management Interface for ACSLS or Oasis RLM, see *Integrating with robot managers* later in this chapter. If you are using SCH:Robot, see *Integrating with SCH:Robot* later in this chapter. If you are using another robot manager, see the documentation supplied with the robot manager.

   Alternatively, you can add tapes manually via the CLI commands ADD and COPY, or via the GUI. New tapes are normally added with status BLANK and then initialized.

   See *Manipulating tape objects* in Chapter 8 for detailed information about tape objects.

8 Initialize tapes for use. If the tapes are to be used by a HYPERtape Backup Node, they must have the correct labels. You can perform this initialization by running the HYPERtape initialization procedure on the OpenMedia client running the Backup Node software. The initialization procedure is a command file called Htape_init.bat for Windows NT systems, htape_init.sh for UNIX systems or HTAPE_INIT.COM for OpenVMS systems. See *Initializing tapes for use with HYPERtape* below for details.

Integrating with robot managers

Proceed as follows to integrate OpenMedia with HYPERtape Robot Manager (HTRM) or HYPERtape Robotic Management Interfaces for ACSLS, Oasis RLM and so on. For information about integrating with SCH:Robot, see the next section.

1 Ensure that the appropriate robot manager is installed (follow the manufacturers instructions). If you are using HTRM, you require version 1.1 or later: the readme.txt file supplied with HTRM includes instructions on how to install HTRM and how to configure the robot.

2 Set up an appropriate tape template called *jukeboxname*-tape.template in the data subdirectory of the OpenMedia top level directory. This file provides basic information for OpenMedia about the types of tape in the jukebox and is used when a robot manager reconcile process needs to add a tape to the OpenMedia database.

   A sample template file called template_jukebox1-tape.template is provided in the bin subdirectory of the OpenMedia top level directory. Copy and edit this file to create an appropriate template for your own jukebox.

3 Add the tapes in the jukebox to the OpenMedia database by running the command file Jukebox_reconcile_*robottype*.bat for Windows NT systems, jukebox_reconcile_*robottype*.sh for UNIX systems, or JUKEBOX_RECONCILE_*robottype*.COM for OpenVMS as appropriate.

   The command file can be found in the bin subdirectory of the OpenMedia top level directory. It interrogates the robot manager to find the contents of the robots and reconcile them with the OpenMedia database. It:

   - Adds tapes to the database if they are in the robot but not in the database
   - Changes the media location to that of the jukebox for existing tapes that have been put into the robot
   - Changes the media location to the dropped media location for tapes that have been removed from the jukebox

   To execute the command file, enter:

   `jukebox_reconcile_`*`robottype jukeboxname`*

   for example:

   `jukebox_reconcile_htrm saturn`

Warning	*This procedure must be run whenever tapes are added to the jukebox or removed from it.* *The default template file adds tapes with a BLANK status. These tapes are then candidates for htape_init to initialize for use by HYPERtape. If you add any other tape to the robot, be careful to update the tape object to reflect the correct status. Alternatively, once you have an established system with all tapes initialized, alter the tape template to add tapes with status FOREIGN, to avoid them being accidentally reinitialized.*

Integrating with SCH:Robot

Proceed as follows to integrate OpenMedia with SCH:Robot:

1 Ensure that the appropriate robot manager is installed via the SCH:Robot command:

`robot_install`

2 Ensure that the jukebox, media locations and devices are configured as described in *Configuring the database*, above.

3 The SCH:Robot command:

`robot_config`

sets up an appropriate tape template called *jukeboxname*-tape.template. This file provides basic information for OpenMedia about the types of tape in the jukebox and is used when a robot manager reconcile process needs to add a tape to the OpenMedia database.

The robot_config command queries the OpenMedia database for the information required to create the template file, but it allows you to override the information extracted from OpenMedia.

The SCH:Robot software allows the robot to be reconciled from any client on the network. The master copy of the template file is stored on the SCH:Robot server system and the template file is downloaded into the OpenMedia client system as required.

A sample template file called template_jukebox1-tape.template is provided in the bin subdirectory of the OpenMedia top level directory. You can compare this file with the file generated by SCH:Robot. You can add any field in the tape object into the template file.

3 Add the tapes in the jukebox to the OpenMedia database by running the SCH:Robot command:

`robot_inventory`

This command may be run from any OpenMedia client where the SCH:Robot client software is installed. It interrogates the robot manager to find the contents of the robots and reconcile them with the OpenMedia database. It:

- Adds tapes to the database if they are in the robot but not in the database
- Changes the media location to that of the jukebox for existing tapes that have been put into the robot
- Changes the media location to the dropped media location for tapes that have been removed from the jukebox

Warnings *This procedure must be run whenever tapes are added to the jukebox or removed from it.*

The default TEMPLATE file adds tapes with a BLANK status. These tapes are then candidates for htape_init to initialize for use by HYPERtape. If you add any other tape to the robot, be careful to update the tape object to reflect the correct status. Alternatively, once you have an established system with all tapes initialized, alter the tape template to add tapes with status FOREIGN, to avoid them being accidentally reinitialized.

Initializing tapes for use with HYPERtape

The HYPERtape Backup Node software runs under the Windows NT, UNIX and OpenVMS operating systems. Proceed as follows to initialize tapes for use with this software:

1 Install the Backup Node software if it is not already present on the OpenMedia client.

Note	*For a UNIX Backup Node, you must install version 1.7 or later of the HYPERtape Backup Node software and choose the OpenMedia option.*

OpenMedia maintains a record of where other HYPERtape components reside. For OpenVMS and UNIX systems, this is done dynamically, but for Windows NT systems it is done when OpenMedia starts or continues after being paused. If the Backup Node is newly installed on a Windows NT system, stop and restart or pause and continue the OpenMedia service using the Service Manager in Control Panel, so that its HYPERtape configuration is updated.

2 From the command line or a command prompt window, set the default directory to the OpenMedia bin directory and run the initialization procedure htape_init by entering one of the following commands.

Windows NT:

```
cd %ommhome%\bin
Htape_init jukeboxname [mediatype]
```

OpenVMS:

```
SET DEFAULT OMMROOT:[BIN]
HTAPE_INIT jukeboxname [mediatype]
```

UNIX:

```
cd $ommhome/bin
./htape_init.sh jukeboxname [mediatype]
```

The *mediatype* parameter to the initialization procedure is optional. If you have already set the appropriate media type during jukebox reconciliation, you don't need to specify the media type in the initialization command. If you supply a media type, you must use a valid, case-specific OpenMedia media type name.

3 The initialization procedure initializes all blank tapes first, then any scratch tapes that do not have the correct HYPERtape format for the operating system, for example HTAPE1 for UNIX, HTAPE2 for Windows NT. The format is held in the tape object, see the section *Tape window* in Chapter 8 for details.

4 If the jukebox has more than one drive, you can initialize several tapes in parallel: simply run the command again. You can have as many copies of the initialization procedure running as there are connected drives.

5 The tapes in the jukebox are now ready for use for use by HYPERtape.

Setting up a HYPERtape backup object

During the save operation, the HYPERtape Backup Node uses OpenMedia to choose an appropriate tape, either starting with a new (scratch) tape, or appending the savesets to an existing tape.

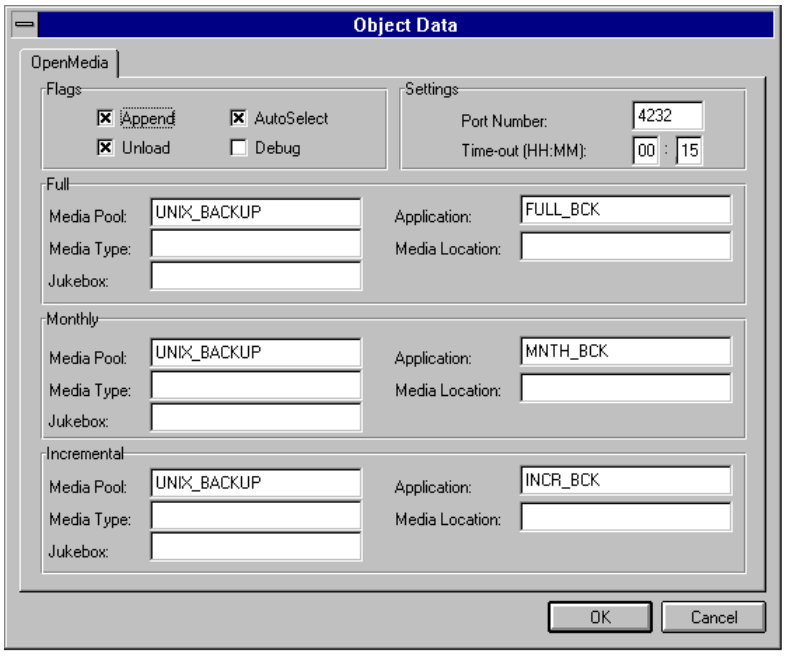
The Backup Node receives information about *backup objects* from the Control Node. This includes information relevant to OpenMedia, such as the saveset name, media information and a number of flags. The following sections explain how to set up the backup object on the HYPERtape Control Node so that OpenMedia can select the tape on which to store the saveset.

Object parameters on a Windows NT Control Node

In order to set up a HYPERtape backup object for use with OpenMedia, you must set specific object parameters in the object database on the HYPERtape Control Node. Use the GUI on the Windows NT Control Node to set the media parameters as follows:

1 Select the Backup Node tab and set the Media indicator to OpenMedia.

2 Click on the Properties button below the list of media indicators to display a window showing OpenMedia fields.



In this window, you must enter the OpenMedia media pools to which the backup tapes for Full, Monthly and Incremental backups will belong. The entries you make must each be the exact name of a valid OpenMedia media pool. No wildcards are allowed.

3 For each of the Full, Monthly and Incremental backups, you must enter an Application. This is used to determine which objects share the same sets of tapes within the media pool. No wildcards are allowed.

4 For each of the Full, Monthly and Incremental backups, you can optionally enter the OpenMedia media type, media location and jukebox to be used. Although these are not necessary, you are strongly recommended to include the media location of the jukebox you are using, to avoid the possibility of OpenMedia considering tapes that cannot be used. Wildcarded entries are allowed in these fields.

5 Set the Unload and Append flags as necessary. The Unload flag governs whether the tape is to be dismounted at the end of the operation. The Append flag determines whether OpenMedia selects a scratch tape for the saveset, or appends the saveset to an existing tape. If the Append flag is set, OpenMedia looks for a suitable tape on which to append the saveset. If a suitable tape is not found, the saveset is written to a scratch tape. If the Append flag is not set, the saveset is written to a scratch tape.

The entries you make are not case-sensitive.

Object parameters on a UNIX Control Node

In order to set up a HYPERtape backup object for use with OpenMedia, you must set specific object parameters in the object database on the HYPERtape Control Node. The media parameters you should set are as follows:

1 Set the Media System field to OPEN_MEDIA.

2 Enter the OpenMedia media pools to which the backups will belong in the Incremental Save Media Type, 1st Full Save Media Type and 2nd Full Save Media Type fields. These entries must match exactly with valid OpenMedia media pools. No wildcards are allowed.

You can optionally append to these fields the names of the media type and jukebox to be used, in the form:

mediapool*[*mediatype*[*jukebox

for example:

FULL_BACKUPS[DLT4000-40GB-C[INFINITY

If you do not include these entries, you should still include the [characters, for example:

FULL_BACKUPS[[

You can specify wildcarded entries for media types and jukebox names.

3 You must enter an Application field in the Incremental Save Media Owner, 1st Full Save Owner and 2nd Full Media Owner fields. This value is used to determine which objects share the same sets of tapes within the media pool. No wildcards are allowed.

You can optionally append to these fields the media location to be used, in the form:

application*[*medialocation

for example:

FULL[ROBOT1

If you do not include these entries, you should still include the [characters, for example:

FULL[

You are strongly recommended to include the media location of the jukebox you are using, to avoid the possibility of OpenMedia considering tapes that cannot be used. You can specify wildcarded entries for the media locations.

4 Set the Dismount and Append flags as required. The Dismount flag governs whether the tape is to be dismounted at the end of the operation. The Append flag determines whether OpenMedia selects a scratch tape for the saveset, or appends the saveset to an existing tape. If the Append flag is set to Yes, OpenMedia looks for a suitable tape on which to append the saveset. If a suitable tape is not found, the saveset is written to a scratch tape. If the Append flag is set to No, the saveset is written to a scratch tape.

Object parameters on an OpenVMS Control Node

In order to set up a HYPERtape backup object for use with OpenMedia, you must set specific object parameters in the object database on the HYPERtape Control Node. The parameters you should set are as follows:

1 In object parameter F66, set the Media manager to OPEN_MEDIA.

2 In the media type subfields of the Full, Monthly and Incremental storage information parameters F67 and F68, enter the OpenMedia media pools to which the backup tapes will belong. These entries must match exactly with valid OpenMedia media pools. No wildcards are allowed.

You can optionally append to these fields the names of the media type and jukebox to be used, in the form:

mediapool*[*mediatype*[*jukebox

for example:

FULL_BACKUPS[DLT4000-40GB-C[INFINITY

If you do not include these entries, you should still include the [characters, for example:

FULL_BACKUPS[[

You can specify wildcarded entries for media types and jukebox names.

3 In the owner subfields of the Full, Monthly and Incremental storage information parameters F67 and F68, you must enter an Application field. This is used to determine which objects share the same sets of tapes within the media pool. No wildcards are allowed.

You can optionally append to these fields the media location to be used, in the form:

application[medialocation

for example:

FULL[ROBOT1

If you do not include these entries, you should still include the [characters, for example:

FULL[

You are strongly recommended to include the media location of the jukebox you are using, to avoid the possibility of OpenMedia considering tapes that cannot be used. You can specify wildcarded entries for the media locations.

4 Set the Dismount and Append flags as appropriate. The Dismount flag governs whether the tape is to be dismounted at the end of the operation. The Append flag determines whether OpenMedia selects a scratch tape for the saveset, or appends the saveset to an existing tape. If the Append flag is set to Yes, OpenMedia looks for a suitable tape on which to append the saveset. If a suitable tape is not found, the saveset is written to a scratch tape. If the Append flag is set to No, the saveset is written to a scratch tape.

Setting up a HYPERtape Backup Node

Backup Nodes are designed to integrate with OpenMedia, and generally require no action. However, you must install UNIX Backup Nodes using the OpenMedia option.

To allow automatic accreditation of the Backup Node during login to OpenMedia, the OpenMedia database must contain a user account object for the Backup Node host. The native username specified in this object must be the username under which the Backup Node software is started. This is usually *SYSTEM* for Windows NT and OpenVMS Backup Node hosts and *root* on UNIX Backup Node hosts.

Removing HYPERtape OpenMedia

The following sections explain how to remove OpenMedia from various supported platforms.

You must stop the OpenMedia service before attempting to remove it from any host.

Windows NT (Intel) hosts

1 Log in as Administrator.

2 Launch the OMM Uninstall icon and follow the instructions. This removes the OpenMedia service, icons, environment variables, registry entries and all files originally delivered.

3 Delete any remaining files and directories in the OpenMedia top level directory.

4 Delete the OpenMedia top level directory.

Windows NT (Alpha) hosts

1 Log in as Administrator

2 Start a new MS-DOS command prompt window

3 Set the default directory to the OpenMedia bin directory by entering:

```
cd /d %ommhome%\bin
```

4 Execute the OpenMedia Uninstall procedure, passing it the name of the .isu file found in the OpenMedia top level directory. For example:

```
uninst -f%ommhome%\DelsL1.isu
```

This removes the OpenMedia service, icons, environment variables, registry entries and all files originally delivered.

5 Delete any remaining files and directories in the OpenMedia top level directory.

6 Delete the OpenMedia top level directory.

Windows 95 hosts

1 Launch the OMM Uninstall icon and follow the instructions. This removes the OpenMedia service, icons, environment variables, registry entries and all files originally delivered.

2 Delete any remaining files and directories in the OpenMedia top level directory.

3 Delete the OpenMedia top level directory.

UNIX hosts

1 Log in as root.

2 Ensure that the OpenMedia bin directory is not in the path.

3 Delete all the files and directories in the OpenMedia top level directory.

4 Delete the OpenMedia top level directory.

5 Delete the files /etc/ommsetup.sh and /etc/ommsetup.csh.

OpenVMS hosts

1 Log in as a privileged user such as SYSTEM.

2 Execute the OMM_REMOVE procedure by entering:

```
@OMMROOT:[BIN]OMM_REMOVE
```

This removes installed images, logical names and all files.

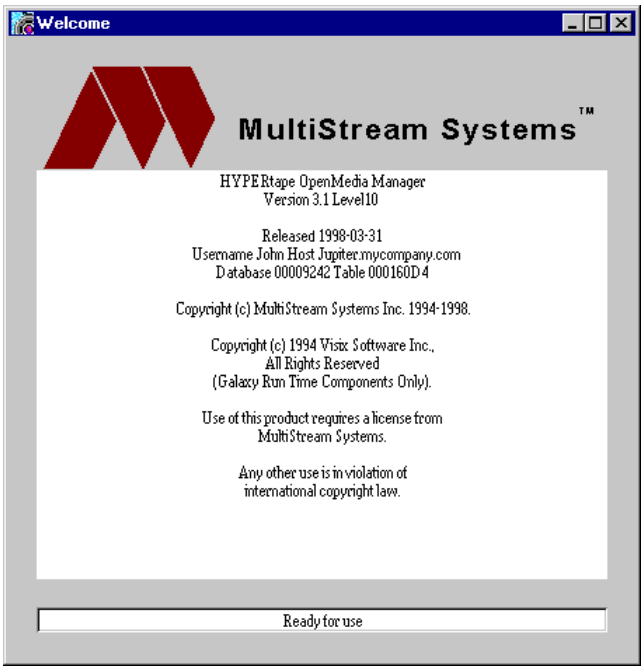
3 Delete the OpenMedia top level directory.

3

Using the GUI

Starting the GUI

When you start the GUI, a Welcome window appears, showing version, copyright and licensing information.



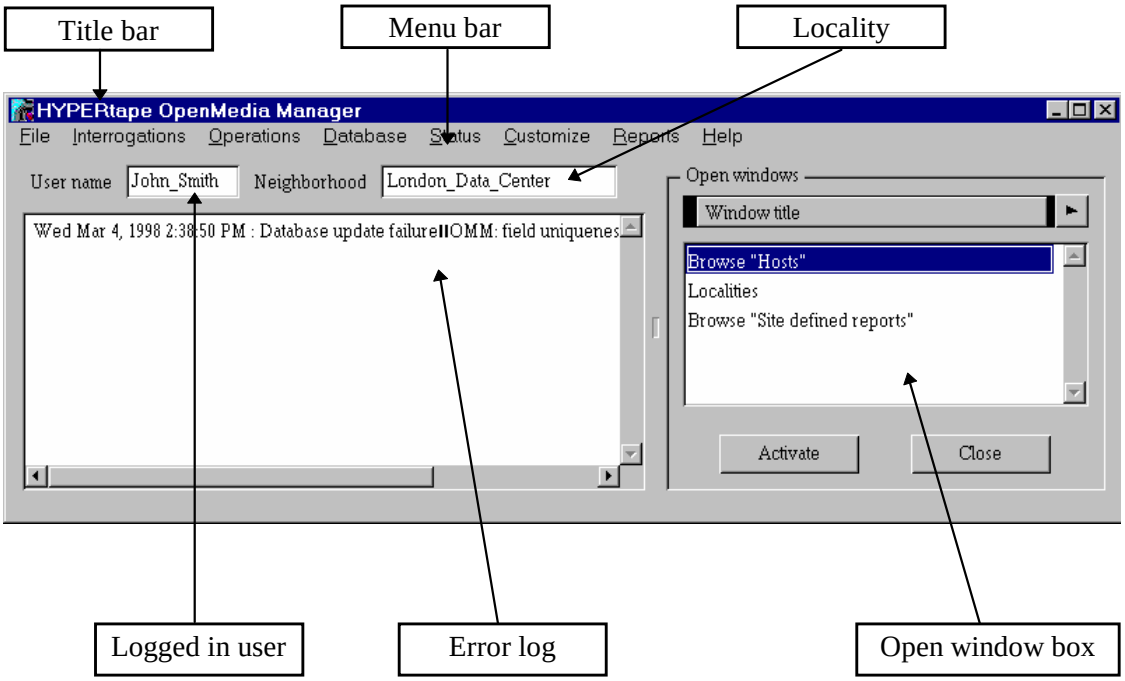
If you need to contact your supplier or local customer support office with a query, you will be asked for the version information displayed on this screen. To redisplay the screen, select the **On version** option on the main window **Help** menu.

While the Welcome window is displayed, OpenMedia attempts an automatic *native accreditation* login. It uses your host name, the username under which you are logged into the host, and information held in the host's local configuration file LOCAL_CONFIG.DAT, which is set up during installation.

After a few seconds, the message line at the bottom of the Welcome window indicates whether the login was successful, and the OpenMedia Main window appears. If the login was successful, this window shows the OpenMedia user name under which you are logged in and the locality to which you are connected.

Main window

The OpenMedia GUI has a set of windows. You access all other windows via the Main window, which appears after the GUI starts.



The *title bar* at the top of the window shows the title *HYPERtape OpenMedia Manager*. Below this is the *menu bar*, showing the names of the menus available.

Below the menu bar are boxes showing the name of the logged in OpenMedia user and the neighborhood locality of the server. These show *Not connected* when the GUI is not connected to the OpenMedia service.

On the left of the main part of the window is the Error log for your OpenMedia session. This details any security violations and error messages displayed by the GUI. To the right of the Error log is the Open windows box, which contains a list of all currently open windows. From this box you can highlight the name of a window and activate it using the **Activate** button or close it using the **Close** button.

You can perform more than one operation at a time using the GUI. For example, you can start a search for tapes that will have expired by a given date and, while the system is searching the database, look up the status of a reserved device. In this case, the Open windows box would show the names of the windows listing expired tapes and reserved devices. For this reason, you never see an hourglass with OpenMedia. You can always start more interrogations, or expand the first object from an interrogation while other objects are still being retrieved.

You can manipulate the size and shape of the Main window, and scroll the Activities log and Open windows list in the normal way for the platform on which you are working.

Note	*The GUI employs industry standard windowing techniques, with standard look-and-feel for all supported platforms. If you are new to windowing systems, you should read the documentation supplied with the operating system and practice for a few minutes before you start to work.*

Menu items

You can access a menu by clicking on its name, or by holding down the Alt key and typing the underlined letter in the menu name. For example, Alt+F displays the **File** menu.

The Main window has the following menus:

File
File operations such as **Login** and **Exit**.

Interrogations
Standard interrogations and reports. Items in this menu provide *selection* windows in which you specify selection criteria. The objects that satisfy these criteria are then displayed in a *browser* window, which you can customize and print as reports. From the browser, you can also select individual objects and view their details.

Operations
Tape and saveset management operations, such as recording movement of tapes onsite and offsite, expiring savesets and removing expired savesets.

Database
Database object maintenance. From this menu, you can access existing database objects via selection and browser windows, or add new database objects from scratch.

Status
Items on this menu allow you to interrogate volatile objects, such as the tapes and devices currently in use.

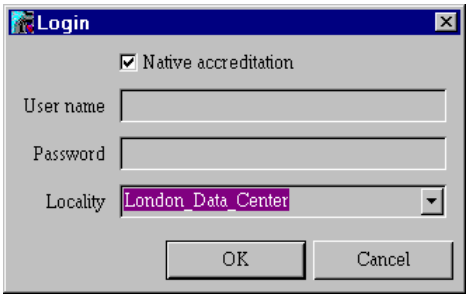
Customize
Individual settings for the GUI, such as font and color preferences.

Reports
Allows you to run and print site specific reports that you have defined using the CLI.

Help
Help topics and version and licensing information.

Logging in

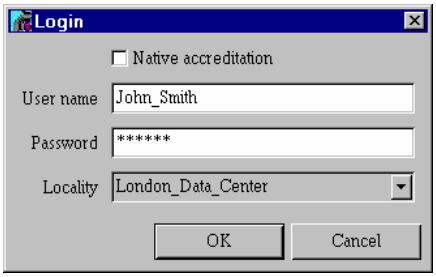
To log in to OpenMedia, for example if you want to log in to a different service from the default used during startup, select **Login** from the **File** menu in the Main window. The Login window appears.



You can log in to the GUI using either *native accreditation* or *password accreditation*. Set or clear the Native accreditation checkbox to indicate the method you are using.

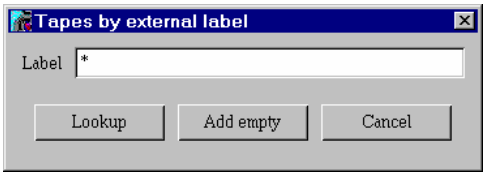
When you use native accreditation, you only need to select the server's neighborhood locality: the server uses a combination of the host name and native username of the client from which you are working to find and check your OpenMedia user name. When you start the GUI, OpenMedia automatically attempts to log you into the locality named in the host's local configuration file, using native accreditation.

If you use password accreditation, you must enter a valid user name and password from an OpenMedia user object, and select the neighborhood locality of the server with which to connect. The characters you type in the password do not appear on the screen, for security reasons.

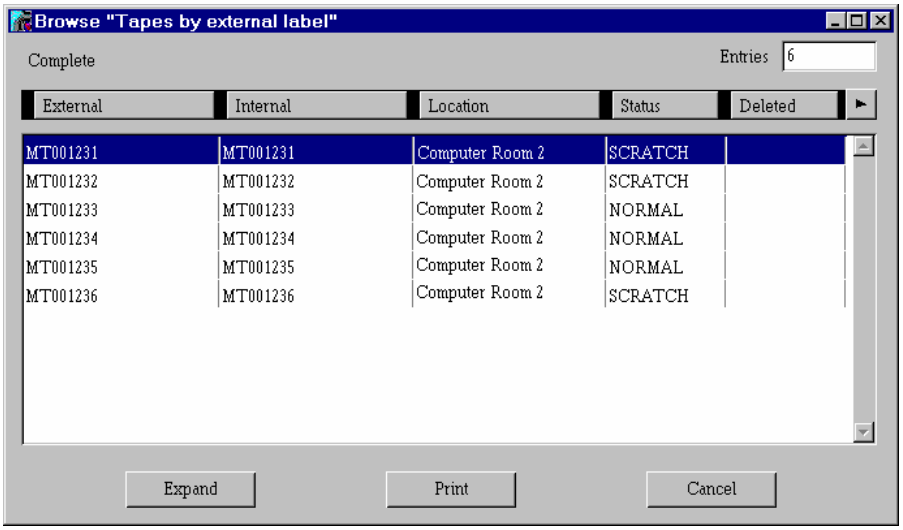


Finding your way round the GUI

Some of the menu options, notably those on the **Interrogations** and **Database** menus, take you through a series of windows. When you select one of these options, a selection window appears. In this window, you can enter wildcarded information about the object or objects in which you are interested, so that OpenMedia can retrieve the object(s) from the database. For information about using wildcards in selection windows, see the section *Using wildcards in selection windows* later in this chapter.



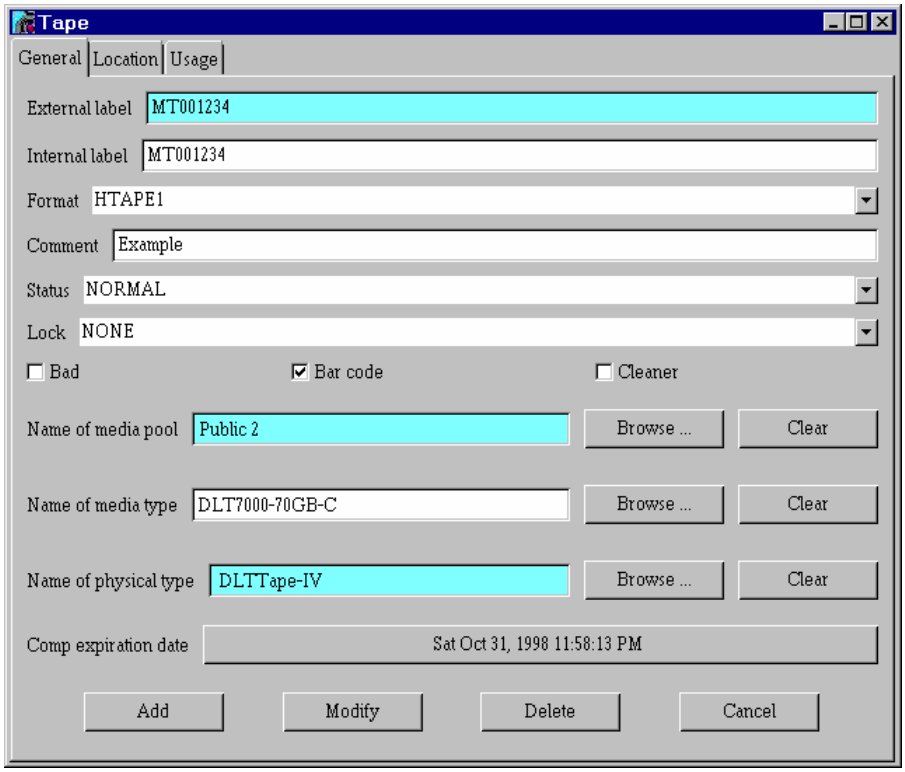
Click on the **Lookup** button to display a list of objects that match your selection criteria. The objects are presented in a browser window.



From a browser window, you can select an object to display in further detail in a *detail* window. Double-click on the item, or click on it once to highlight it and then click on the **Expand** button.

You can also customize and print the browser window. For details, see the section *Printing reports* later in this chapter.

If you select an option from the **Database** menu, you can alternatively go straight from the selection window to a blank detail window by clicking on the **Add empty** button in the selection window.

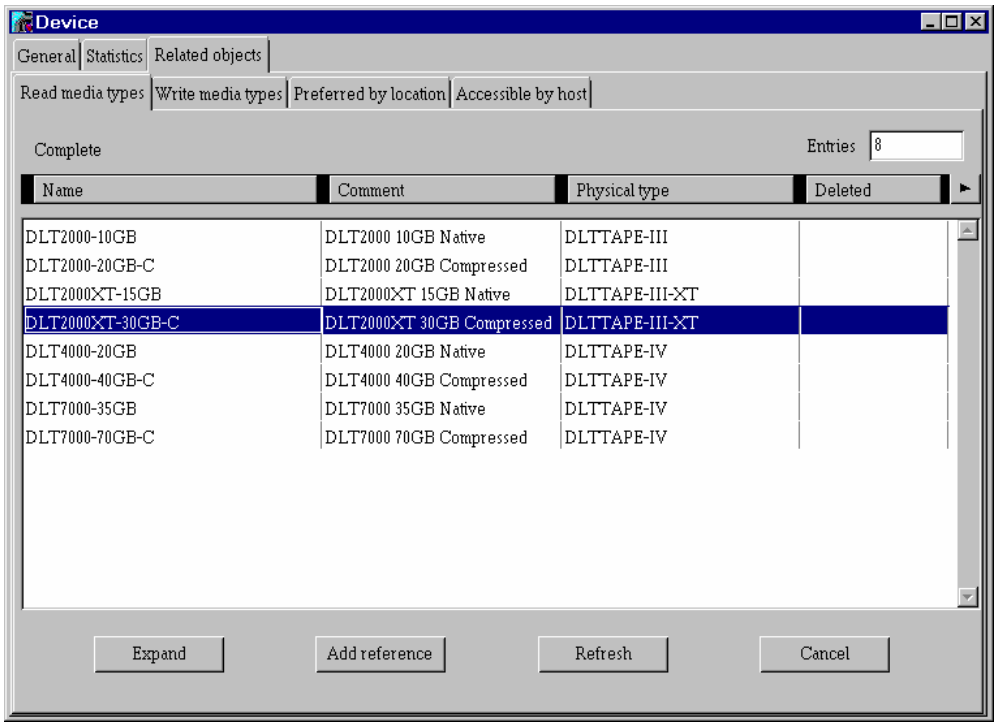


From a detail window, you can add, modify and delete objects, providing you have the correct privileges. For details of these procedures, see the section *Managing database objects* later in this chapter.

Some detail windows carry a large amount of information, so they are divided into a number of *tabs*. For example, a tape detail window has tabs for general, location and usage information. The first tab displayed when the detail window appears is the *General* tab. Click on a tab header to display the tab.

Some detail windows have a *Related objects* tab, which displays lists of other objects to which the current object is related. The Related objects tab may have its own inner tabs, so that each type of relationship is shown in its own list. For example, the Related objects tab of a device detail window has separate inner tabs to show:

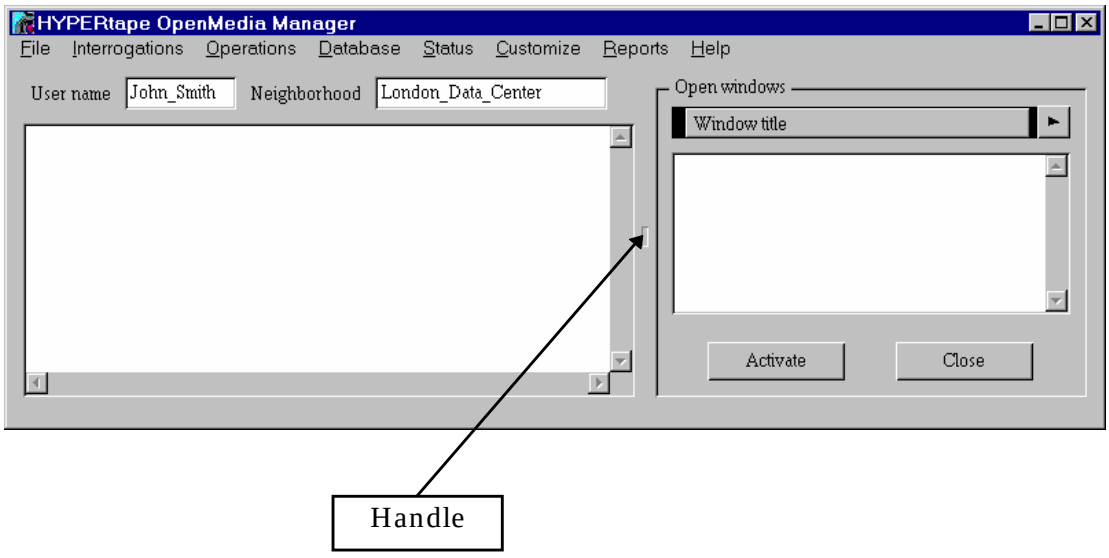
- The media types the device can read
- The media types the device can write
- Any media locations for which this is a preferred device
- The hosts that can access the device.



Note	*If you close a window, all lower level windows associated with it are also closed. For example, if you close the browser window from which you selected a device, the device detail window is also closed.*

Resizing boxes

Some windows have *handles* that allow you to resize the boxes with which they are associated. For example there is a handle on the Main window that allows you to change the width of the area containing the login information and Activities log. To resize the box, click on the icon, hold down the mouse button and drag the icon to the required position.



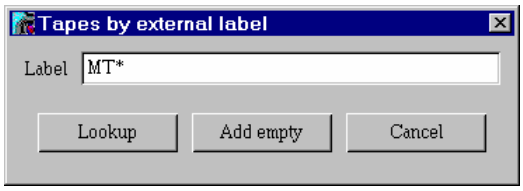
Reducing windows to icons

You can reduce windows to icons. If you reduce a detail window, its associated browser remains; only the detail window is reduced. However, if you reduce the browser window, the detail window is also reduced, and the name on the icon is that of the browser. If you restore a window, all lower level windows associated with it are also restored.

Using wildcards in selection windows

When providing information for search criteria, the more detail you specify to the search, the shorter the resulting list. You can use asterisk (*) and question mark (?) wildcard characters to generate partial lists.

The asterisk (*) wildcard matches zero or more characters. For example, MT* matches MT01231, MT01232, MT01233 and so on.



The question mark (?) wildcard matches a single character. For example, SL?P matches SLIP and SLAP but not SLOOP.

Note	*To produce a list of all possible objects, enter *. You should only leave a field blank if you want to match objects in which that field is blank.*

Using browsers

When you select items from the **Interrogations** and **Database** menus, lists of objects are displayed in browser windows. You can scroll through these lists and choose individual objects for further work.

At the top left of a browser window is a progress indicator field.

The indicator shows:

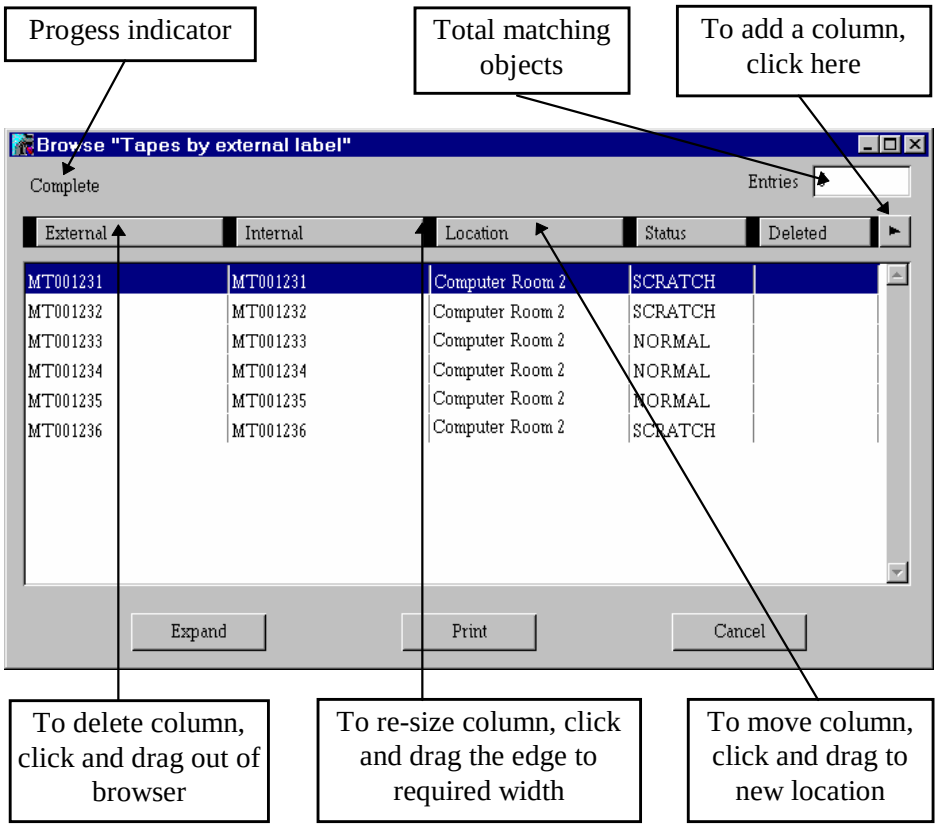
`Working...`

while the database search is in progress, and:

`Complete`

when the search has completed successfully.

To select an object, click on it. The object is highlighted and you can then click on a command button. You can display the object's detail window by highlighting it and clicking on **Expand**. Double clicking on the object name has the same effect.

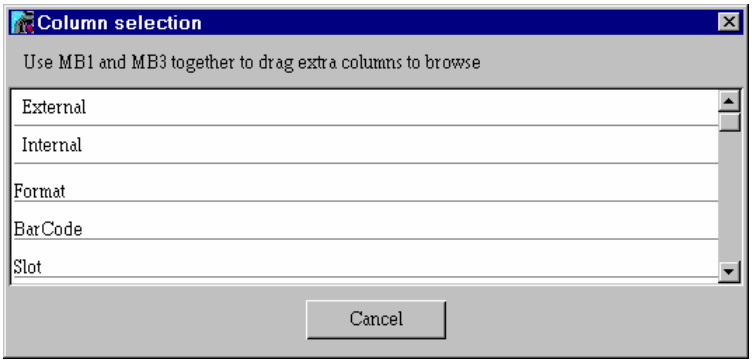


The browser window shows summary information in the form of a number of object fields and a Deleted column. A large red cross in this column indicates that the object has been deleted from the database. You can reinstate the object by expanding it and adding it back, but once you have left the browser you cannot recover a deleted object.

You can customize a browser window by adding, moving, resizing and deleting columns (see below for details). These changes are temporary: the next time you use the browser, the layout reverts to the default. However, these facilities are useful if you want to produce a report from the browser. Once you have made the changes, simply click on **Print** and proceed as described in the section *Printing reports* later in this chapter.

Adding columns

To add a column, click on the arrow button on the right of the column title bar to display a list of the columns you can include in the browser.



You can then *drag and drop* columns from the list to browser's column heading bar using the technique described in the section *Drag and drop technique* later in this chapter.

Moving columns

To move a column within the browser, you simply click on the column name, hold down the mouse button and drag the column name to the position you require.

Resizing columns

To change the size of a column, click on the side of the box containing the column name, hold down the mouse button and drag the box to the required width.

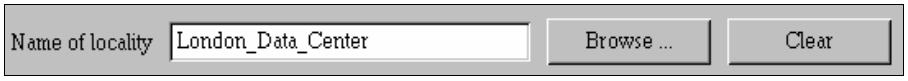
Deleting columns

To delete a column, simply click on the column name, hold down the mouse button and drag the column outside the browser.

Entering data into detail windows

Setting object name fields

Some windows contain references to related objects. For example, a host object must refer to a locality.



These relationships are represented by fields in the object, which you can only set by dragging the name of the related object from a browser window using the technique described in the *section Drag and drop technique* later in this chapter. For your convenience, all such fields are associated with a **Browse...** button, which you click on to display a browser window showing the possible related objects. To clear the field, click on the associated **Clear** button.

Setting counter fields

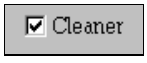
Some windows contain numeric fields with pairs of arrow buttons on the right-hand side. For example, you can set the number of errors that initiate automatic cleaning of a device in a robotic jukebox.



To set a counter field, use the up arrow button to increase the number in the field, or the down arrow button to decrease the number in the field. You can also enter a number directly into the field. OpenMedia ignores any non-numeric entry and retains the previous value in the field.

Setting checkboxes

A checkbox field is a field that can have only two states: *set* or *clear*. For example, a tape can be locked from use or not locked. These fields are represented in the GUI by checkboxes.



To set or clear a checkbox, click on it.

Setting dates and times

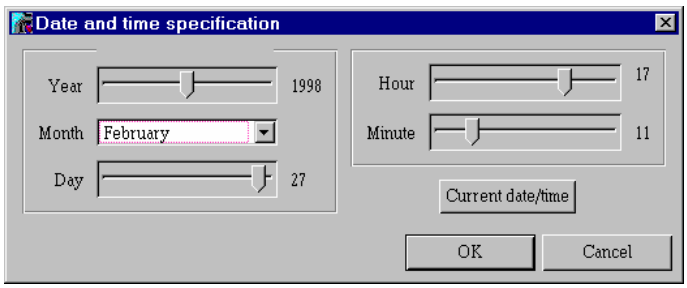
OpenMedia uses two types of field to measure time:

- *Date* fields hold specific dates and times such as the date a saveset was created
- *Deltatime* fields hold intervals of time such as the default offsite duration in a media pool object

Both types of fields are displayed in the GUI as buttons. They can appear in both detail windows and selection windows. If the field is blank, the button shows *No date set* or *No deltatime set* as appropriate.

To set one of these fields, click on its button to display either the Date and time specification window, or the Deltatime specification window, whichever is appropriate.

Date and time specification window



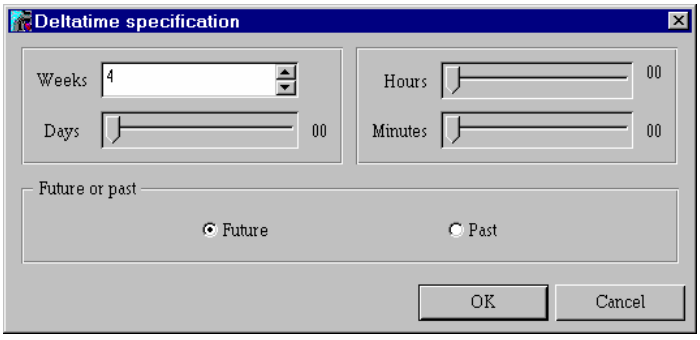
In the Date and time specification window, you set the year, day, hours and minutes that you require, within a time span of 100 years from the initial date set in the window using *sliders*, which have slider icons that can move along a line. Click on the slider icon, hold down the mouse button and drag the slider to the right or left until the value you require is displayed next to the slider.

Alternatively, you can click to the left or right of the slider icon. Click once to increase or decrease the value by a single unit, or hold down the mouse button to change the value continuously until the slider icon reaches the position of the mouse pointer.

You select the month from a pull-down list.

To select the current date and time, click on **Current date/time**.

Deltatime specification window



When you set a deltatime field, you enter a period of time measured in weeks, days, hours and minutes. In selection windows, this is based on the current time when you made the entry. For example, a value of 3 in the weeks field indicates a date three weeks from the time of entry.

In the Deltatime specification window, you set the weeks value by entering a number or clicking on the counter icons at the right of the field to increase or decrease the current value. You set the days, hours and minutes values using sliders.

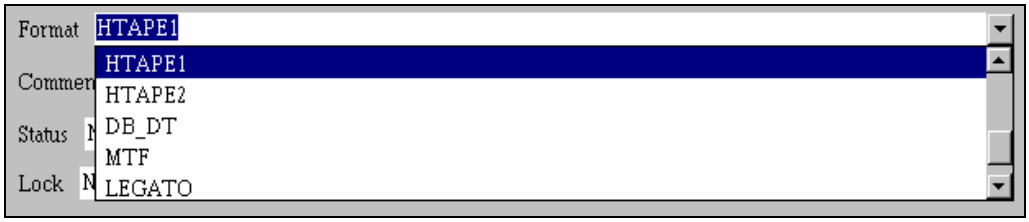
The Deltatime specification window also has a pair of indicators labeled future and past. These are for use when you are specifying a time interval in a selection window, when the interval you set is measured relative to the current time. For example, you might be interested in events that occurred yesterday, or one week ago, in which case set the past indicator. Or you might want to know if something is due to happen tomorrow or in a week's time, in which case you would set the future indicator. In detail windows, you should always leave the future indicator set (you cannot set both).

Using pull down lists

When the value in a field must be one of a preset range, the field is represented in the GUI with an arrow button on the right-hand side. For example, the format of a tape must be a recognized keyword.



To set such a field, click on the arrow button to display a list showing all the acceptable values, and select one by clicking on it.



If the list is too long to display in full, a scroll bar allows you to navigate the list.

Drag and drop technique

Drag and drop is the technique of using the mouse to move the name of an object in a browser window to a field in a detail window. The technique is specific to OpenMedia, and is used to create links between database objects. The procedure is as follows:

1 Locate and highlight the object in a browser window.

2 Place the mouse pointer over the object.

3 If your GUI has OPENLOOK look-and-feel, press and hold the Ctrl key and mouse button 1 (MB1) and drag the object name to the new location.

If your GUI has Motif or Windows look-and-feel, press and hold both the left and right mouse buttons and drag the object name to the new location.

A red target icon appears as you drag the object. This changes to a green bulls eye when the object is correctly located over the target.

4 Release the mouse buttons.

Note	*On screens with low resolution, you may have to resize or move windows to expose the target of the drag and drop.*

Standard command buttons

Many windows have *command buttons*, which you click on to perform actions such as **Cancel** or **Modify**. These buttons are usually situated at the bottom of the window, or at the bottom of the area to which they apply if they do not apply to the whole window. If a command button is grayed out it is not available, and clicking on it is ineffective. The main command buttons are:

Activate
Activates the highlighted window.

Add
Adds a new database object, such as a new tape. If you are adding an object with similar details to an existing object, you can display and alter the details of the existing one, then use **Add** to create a new database object containing the altered information.

Note	*The add button only adds a new object; it does not relate the new object to others unless the object is a Device object. For information about relating objects, see the section **Adding related objects** later in the chapter.*

Add empty
This button appears on selection windows and displays an empty detail window so that you can add a new database object.

Add reference
Some object detail windows have Related object tabs, which display lists of other objects that are related to the current object. For example, a host is related to the devices it can access and a device is related to the hosts that can access it. The **Add reference** button on a Related objects tab allows you to add a new reference to the list.

Browse...
Fields with an associated **Browse...** button contain references to related objects. The only way you can set these fields is by inserting the name of the related object from a browser window using the technique described in the section *Drag and drop technique* earlier in this chapter. Click on the **Browse...** button to display the browser window showing the possible related objects. Fields with an associated **Browse...** button also have a **Clear** button to clear the field.

Cancel/Close
Exits from the current window without making any changes.

Clear
Clears a field associated with a **Browse...** button.

Current date/time
Sets the current date and time in a field, via the date and time specification window.

Delete
Deletes the current database object. If you accidentally delete an object, it is displayed in the browser window with a red cross in the Deleted column. You can reinstate the object by reselecting it from the browser and clicking on **Add** in the detail window.

Note	*OpenMedia will not allow you to delete an object that is related to other objects. For example, you cannot delete a media location while there are tapes in it.*

Expand
Displays the detail window for the selected object. For example, after creating a list of tapes and highlighting the one you are interested in, click on **Expand** to display complete details about that tape. Double clicking on the object has the same effect.

Lookup
Starts a search from a selection window.

Modify
Modifies a database object with the changes you have entered and closes the window.

OK
Confirms your entries so that any appropriate action can take place and closes the window.

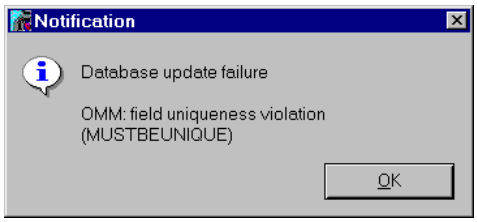
Print
Prints a report of the list in a browser window.

Refresh
Updates the list in the Related object tabs of object detail windows when references are added or deleted.

Run Report
Runs a report from the Reports window

Notification boxes

If OpenMedia detects an error, it displays a *notification box* giving an explanation of the problem.



Click on **OK** to acknowledge that you have seen the message and to make the window disappear, or you may have to answer **Yes** to retry an operation or **No** to cancel it. For a list of OpenMedia messages with explanations, see Appendix B.

Getting help

OpenMedia has a comprehensive help system providing most of the information contained in this manual. You can access help from the **Product information** option on the **Help** menu, which displays a window showing the help contents list. You can locate topics using the contents list, or by searching for specific keywords using the **Search** option on the help **Navigate** menu.

Context sensitive help is also available on most windows of the GUI. Press the HELP key for your platform (e.g. F1 on Windows NT) to display help about the field or window in which the mouse pointer is positioned.

Further information about using online help is available from the help system itself.

To display the Welcome window, which is shown when you start the GUI, select **On version** from the **Help** menu. This window contains information about the software version, which you will need to quote if you contact your supplier or local support office with a query.

Managing database objects

In general, you can carry out the following tasks for all database objects, providing you have the correct privileges:

- List existing database objects and, from the list, select objects for modification or deletion
- Add new database objects
- Modify existing database objects
- Delete database objects
- Print reports

Listing database objects

To display a list of database objects, select the appropriate option from the **Database** menu. A selection window appears, in which you can enter selection criteria for the object(s) in which you are interested. You can use the * and ? wildcards in your entries. When you have entered the necessary details, click on **Lookup** to display a browser window containing the objects that satisfy the selection criteria. For example, if you enter B* in the label field of a Tapes selection window, the browser list contains all the tapes whose label begins with B.

From the browser window, you can select an individual database object to view, modify or delete. You can also customize the browser and add, remove or rearrange its columns using the techniques described in the section *Using browsers* earlier in this chapter. Then you can print the result as a report.

Adding database objects

To create a new database object from scratch, select the appropriate option from the **Database** menu. To display a blank screen on which to enter the details of the new object, click on **Add empty**. Alternatively, you can add a database object by modifying the details of an existing object. Select and modify the object, then change its name and click on **Add**.

Some objects contain mandatory fields that specify the name of another object type. For example, a jukebox requires a locality, and a media type requires a physical type. These dependencies may affect the order in which you choose to create new objects. For example, you should check if the relevant physical is in the database before you create a media type object that refers to it. The dependencies for each object type are:

Locality
None

Jukebox
Requires a neighborhood locality

Media location
Requires a neighborhood locality

Device
Requires a neighborhood locality and jukebox

Host
Requires a neighborhood locality

User
Requires a domain locality

User account
Requires a domain locality and user

Privilege
Requires a neighborhood locality and user

System policy
Requires a neighborhood locality

Media pool
Requires a domain locality

Physical type
Requires a domain locality

Media type
Requires a domain locality and physical type

Tape
Requires a media location, physical type and media pool

Saveset
None

Segment
Requires a tape

Note	*When adding tapes to a robotic jukebox, you should use the jukebox reconciliation facilities to add the tape objects to the database rather than adding the objects manually. See* ***Configuring HYPERtape OpenMedia*** *in Chapter 2 for details.*

Adding related objects

Relationships between objects are stored in the database two ways, which depend on whether single or multiple relationships are allowed.

Relating single objects

When an object can be related to only one object of a particular type at a time, the related object's identity is stored in a field in the object. An example of this is the locality to which a host relates. You set the field in the object by inserting the related object's name from a browser window. For convenience, all such fields are associated with **Browse...** buttons which bring up the appropriate browser.

To add such a reference, click on the **Browse...** button to display a selection and browser window for the type of object required, click on the related object and insert it into the field using the technique described in the section *Drag and drop technique*.

If, when using this technique to add a reference to an object, you find that the referenced object does not exist, you can create it by expanding another object in the browser, amending it as required and then clicking on **Add**. The new object will then appear at the bottom of the browser and you can drag and drop it into the original object.

Relating multiple objects

Relationships such as those between hosts and the devices they can access, where more than one relationship can exist at a time, are stored in the database as lists of related objects. They appear in the Related objects tabs of the detail windows. To add such a reference, select and display the detail window of the object in the usual way and select the Related objects tab. If there is more than one type of related object, display the type in which you are interested, then click on the **Add reference** button at the bottom of the tab. A detail window appears, showing the original object and a field for the related object. Click on **Browse...** and insert the related object into the field using the technique described in the section *Drag and drop technique*, then click on **Add** to complete the operation.

Example

To add an Accessible by host relationship to a device:

1 Browse devices and select the required device.

2 Select the Accessible by host tab from the Related objects tab. The Accessible by host tab shows any hosts that can currently access the device.

3 Click on **Add reference** to display a detail window showing the device name and an empty field for the name of the host.

4 Click on the **Browse...** button next to the Host field to display the Hosts browser window.

5 Click on the name of the required host in the browser window, then drag it from the browser and drop it into the Host field using the technique described in the section *Drag and drop technique* earlier in this chapter.

6 Click on **Add** to complete the operation.

You can delete a related object by selecting its reference in the list, expanding it and clicking on **Delete**. To modify a reference, you must delete it and add a new one.

When you return to the Related objects tab, click on **Refresh** to update the list with the changes you have made.

Modifying database objects

To modify a database object, select the appropriate option such as **Database/Users**. Display the browser window and select the object, make any changes you require in the detail window, then click on **Modify**.

Deleting database objects

To delete a database object, select the appropriate option such as **Database/Tapes**. Display the browser window and select the object, then, when the detail window appears, click on **Delete**.

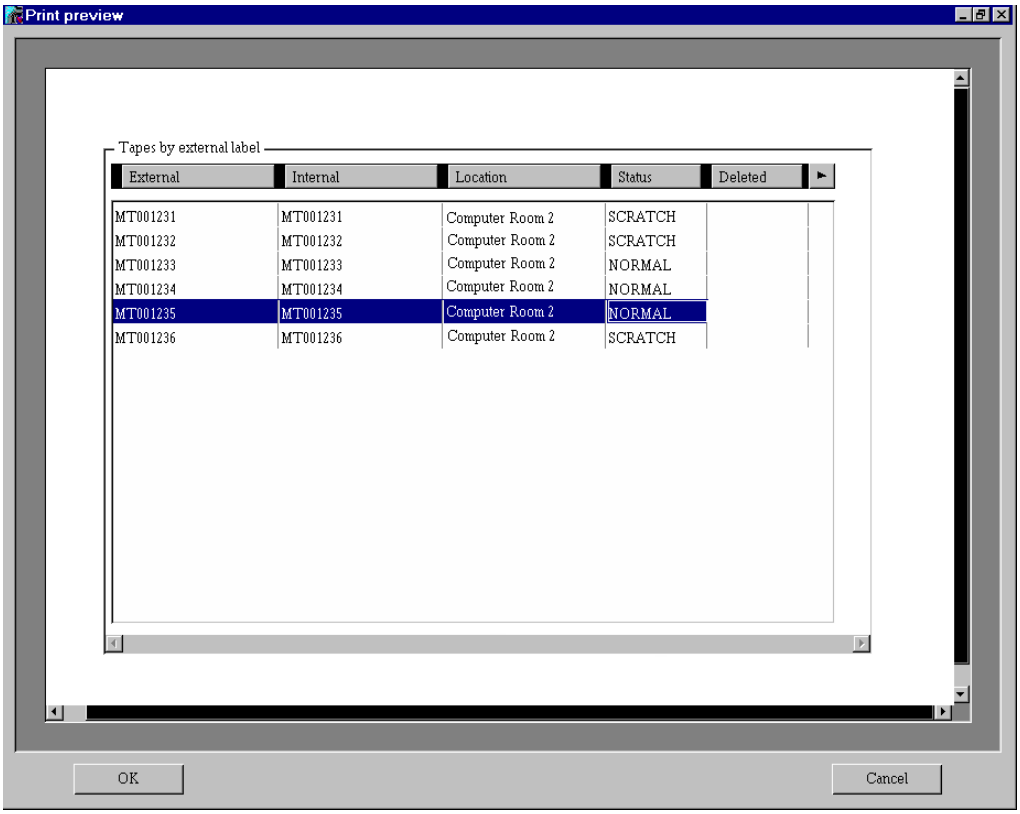
When you click on **Delete**, a warning window appears and you must confirm the deletion before it proceeds.

When an object is deleted, it stays in the browser window, but the Deleted column in the browser window displays a large red cross. If you accidentally delete an object and notice before you leave the browser window, you can redisplay the detail window and click on **Add**.

Printing reports

To print a report from a browser window, click on **Print** in the browser window. If your operating system supports printer selection, a printer selection window appears. Otherwise, the default printer is selected automatically.

When the printer is selected, the Print preview window appears, showing how the report will look when printed.



In this window, the white area represents the paper upon which the report will be printed, the size of which is determined by the paper size set for the printer.

Use the scroll bars at the edges of the paper to view the entire page if it is not all visible.

If the report is too wide to fit on the paper, a third scroll bar appears at the bottom of the report, within the paper margin. You can use this scroll bar to view the entire report.

You can alter the size of the columns, and add or remove columns in the same way as you alter the browser window. For example, if the report is too wide to fit on the paper, you can reduce column widths and remove less important columns or rearrange the columns so that the more important ones are in view.

When you are happy with the report, click on **OK** to print it.

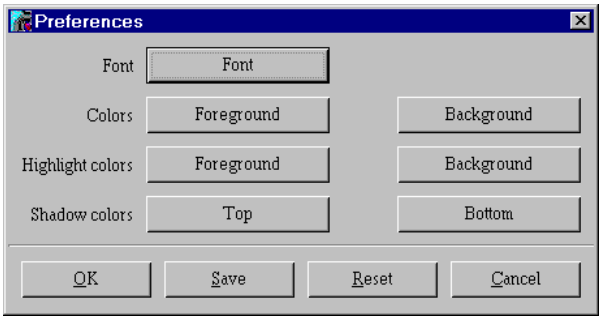
The GUI has a facility for running site defined reports from the **Reports** menu, see *Running site defined reports* in Chapter 8 for details. When you run a site defined report, the GUI displays the results in a browser window and you can print it using the procedure described above.

Setting preferences

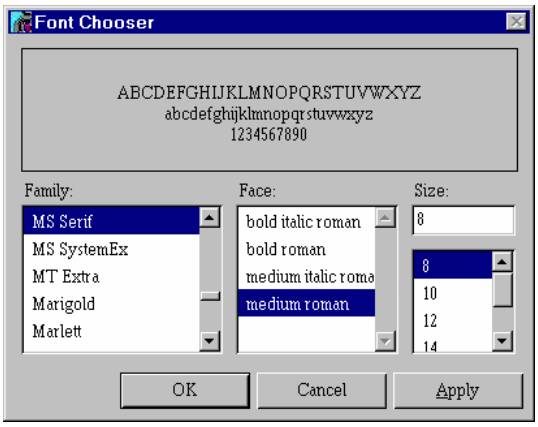
You can change the following settings that determine how OpenMedia is displayed on your system:

- The window colors
- The window fonts

These preference settings are all available as options on the window that appears when you select the **Customize/Preferences** option.



Click on **Font** to display the Font window, from which you can select the font to be used in the GUI windows.



Click on the appropriate color button to customize the foreground, background and shadow colors that appear in the GUI windows.

Note	*The window colors only apply to the Motif or OPENLOOK interface styles. Windows look-and-feel follows the Windows color scheme.*

Leaving the GUI

When you have finished using OpenMedia, select **Exit** from the **File** menu in the Main window. This option closes any OpenMedia windows that remain open and then exits from the GUI.

4

Using the CLI

Introducing the CLI

The CLI is a client application supplied as part of OpenMedia. Its primary purpose is to allow the development of scripts for managing media and allowing the integration of OpenMedia with other tape applications. It can also be useful for remote support operations over dial up modems using connections such as telnet.

The CLI is not the primary user interface to OpenMedia. In general, the GUI provides the higher level user interface, while the CLI is closer to the lower level objects and the internal API.

Each CLI command is associated with an OpenMedia API request. For example, the SHOW command issues a Lookup request on an object.

Information is passed to the request via parameters and options (qualifiers) to the commands. Two syntaxes are available on all supported platforms: *slash syntax* has / (slash) options as used in the DOS and OpenVMS operating systems, while *dash syntax* has - (dash) options as used in UNIX systems.

CLI architecture

The CLI consists of a front end process, which handles commands, and a back end process, which establishes and maintains the connection with the server. This allows you to run a series of commands with a single connection.

CLI command categories

The CLI commands are categorized listed below, with brief explanations of their functions. The commands are listed and explained in more detail in the section *CLI commands* later in this chapter.

Commands that interrogate or manipulate database objects

CLI commands that look up database objects can display information or write it to a file. ADD and SET commands work either from the command line or from a file. The files must contain field names and values that are suitable for manipulation by a simple text editor or script.

The following commands are used to manipulate database objects and interrogate volatile objects such as reserved tapes and reserved devices, providing a network wide interface to the database.

Command	Description
display	Shows the structure of an object
show	Displays selected objects
set	Updates fields in selected objects
add	Adds one or more objects
delete	Deletes selected objects
copy	Adds one or more objects based on an existing object
assign	Defines relationships between objects
deassign	Removes relationships between objects
report	runs a report on selected objects

Commands that manage media objects

The following commands are used to manage media objects and should be used in preference to a raw SHOW and SET commands.

You can use an identifier called a context with some of these commands to identify a particular volatile object such as a reserved tape, reserved device or opened saveset. If you only plan to use a single tape or device at a time, you can ignore contexts.

Command	Description
reserve	Reserves a tape (and optionally a device) for use
unload	Unloads a tape from a device
release	Releases a reserved tape or reserved device
note tape	Notes that a tape has been initialized
scratch	Scratches one or more expired tapes
move	Notes that selected tapes have changed media location or pool
open saveset	Creates and opens a saveset
close saveset	Closes an opened saveset
open segment	Creates and opens a segment on a tape
close segment	Closes an opened segment
expire saveset	Expires selected savesets
remove saveset	Removes selected savesets from the database
status	Displays the status of volatile objects
note device	Notes selected devices have been cleaned or serviced

Commands that process prompts and changes

Prompts provide the mechanism for requesting intervention and waiting for a response. Supported prompts include those to load a tape, unload a tape and messages.

Changes provide the mechanism that allows clients to declare an interest in data objects in the server. If you register for changes then monitor the changes, you can track all the changes that are made to database objects.

The following commands are available for processing prompts and changes:

Command	Description
register	Registers to view prompts or changes to the database
deregister	Reverses a register command
status prompts	Shows any outstanding prompts
reply	Replies to an outstanding prompt
request	Raises a prompt and waits for a reply
monitor	Monitors prompts or changes

Commands that make server requests

The following commands make specific server requests.

Command	Description
server trace	Sets the trace level of the server
server shutdown	Shuts down the server
server database	Instructs the server to access or deaccess the database

Miscellaneous commands

The following miscellaneous commands are available:

Command	Description
login	Logs in to a neighborhood locality
end	Exits both the front and back ends of the CLI
version	Displays CLI version information
mode	Sets or displays CLI defaults
help	Displays help information on commands and topics
quit	Exits the front end of the CLI
exit	Exits the front end of the CLI
echo	Turns command echo on or off for file input

Starting and stopping the CLI

You invoke the CLI from the command line by issuing the command:

`mmcli [`*`command`*`]`

If a command is specified, it is executed. If the CLI is invoked without a command, it operates *interactively*: the CLI writes a prompt and waits for a command to be entered.

When you start the CLI, it automatically starts the front end process and connects to the back end process if it is already running. If the back end process is not already running, it is started and the front end is connected to it.

When you start the CLI back end process, an implicit login is attempted using the name by which the host is known to the network and the username you used to log in to the host. The implicit login is not immediate: it is delayed until a command is issued which requires a login. Some commands can operate without a login, for example, DISPLAY, HELP, END, EXIT, VERSION.

The implicit login only succeeds if the OpenMedia database has a suitable user account object linking your OpenMedia user to the host. If this login is unsuccessful, you must log in using a valid OpenMedia user name and password before you can carry out any database operations. See the section on the LOGIN command for details.

There are two ways of stopping the CLI:

- Use the QUIT or its equivalent EXIT command to quit the front end process, leaving the back end running
- Use the END command to quit both the front end and back end processes

You can use QUIT or EXIT to quit the CLI temporarily, retaining the current login and leaving tapes and devices reserved. You cannot use END until all processing is complete. END logs you out: the next time you start the CLI, you must log in again.

General syntax

A CLI command is made up of a command verb followed by *parameters* and *options* (*qualifiers*). The command verb must be the first element on the command line.

Parameters to CLI commands can be keywords, numbers or selection criteria. For example, in the command:

```
mode limit 5
```

which sets the default limit for database altering commands to 5 objects, *mode* is the command verb, *limit* is a keyword parameter and *5* is a numeric parameter.

Options are identified in either slash or dash style and may be placed anywhere on the command line after the command verb, except in the middle of multi-element parameters such as selection criteria. The command:

```
show tape status.eq.scratch format.eq.ansi_ascii /name
```

displays the names of all tapes with a status of scratch and a format of ANSI ASCII. In this command, *show* is the command verb, *tape status.eq.scratch format.eq.ansi_ascii* are selection criteria parameters and *name* is an option. The above example shows the command using slash syntax for the NAME option. The dash syntax for the command is:

```
show tape status.eq.scratch format.eq.ansi_ascii -name
```

Notes

Options may precede or follow parameters, but may not be embedded in selection criteria. For example, show /name tape status.eq.scratch format.eq.ansi_ascii is valid, but show tape /name status.eq.scratch format.eq.ansi_ascii is invalid.

*Options can be abbreviated (see **Using abbreviations** later in this chapter). The resulting dash letter syntax looks like UNIX, but you cannot specify options -a, -b and -c as "-abc": you must use "-a -b -c".*

Some slash form options use the equals sign (=). This character is interchangeable with the colon (:) on platforms that support the form. However, you are advised to use the colon on Microsoft Windows-based platforms, as bat files on some Windows NT systems treat the equals sign as a space unless the entry is enclosed in quotes.

Further information about selection criteria is given later in this chapter.

CLI Commands may be continued on another line by ending the line to be continued with a dash or a backslash. This character must be followed immediately by the end of line, with no trailing spaces.

Getting help

Full reference information about the format of individual commands, including the parameters required and the options available, is given in the CLI help text. To access this help from the CLI command line, enter HELP, followed by the command about which you require information. For example, enter:

```
help add
```

to display help text about the ADD command, which adds an object to the database.

Help is shown in either slash or dash form, depending on the syntax last used. If you want to specify the form of help, use the MODE option to the HELP command. For example:

```
help add /mode
```

shows help in slash form, while:

```
help add -mode
```

shows help in dash form.

Note	*All CLI entries are case insensitive in both slash and dash forms, except for text values enclosed in quotes (" " or ' '). Selection using quoted strings is case sensitive and generally more efficient, as the database can perform keyed string lookups only if a case sensitive string is given.*

CLI help covers a number of useful topics in addition to that provided for individual commands. To see a list of all the commands and topics covered by CLI help, simply enter:

```
help
```

at the CLI prompt.

Using the DISPLAY command

The DISPLAY command shows details of a specified object type. It shows the structure of the object type in terms of its field names and keys, not the values set in a particular object. The data type of each field is also shown. See the section on *Data types* for further information. Any fields with the annotation ND are not held in the database, and cannot be accessed via the CLI.

If you specify the object type as * (asterisk) all object types are displayed.

If you do not specify an object type, the response to the command shows the names of all object types.

This command is available for both database objects and volatile objects. It is useful if you are unsure of the name of a field you wish to include in another command.

You can optionally use the FILE option to direct output to a file, and display fields in the object header by using the option HEADER.

Using abbreviations

You can abbreviate all CLI command verbs, parameters and options when using the CLI, which can save time and effort when entering commands via the command line. However, you are strongly advised not to abbreviate entries in command scripts, to avoid the possibility of ambiguity arising if new keywords are introduced at later releases of OpenMedia.

When using abbreviations with the CLI, you must avoid ambiguity. This rule includes object types and field names. When a supplied name matches exactly, the exact match is used even if it is an abbreviation of another name.

For example, to specify the object types device and devicestatus:

dev
: is not allowed as it is ambiguous

device
: specifies device, as it matches exactly

devices
: is a valid abbreviation for devicestatus

Field names must be unique within the object, but do not need to be unique across all objects.

Selection criteria

You can use the CLI to interrogate the database, for example, to display details of all the tapes with an error count greater than a certain number. In this case, you would specify a SHOW command, with selection criteria operating on the error count field name to identify the tape objects in which you are interested. Selection criteria for volatile objects are ignored.

Selection criteria take the following format:

object_type [*field_name*.*operator*.*value*]...

where the operator is one of the following:

EQ	(equal to)
GT	(greater than)
LT	(less than)
LIKE	(wildcarded equal to)
BTW	(between)
NE	(not equal to)
GE	(greater than or equal to)
LE	(less than or equal to)
UNLIKE	(wildcarded not equal to)
OUT	(outside)

Operators can be in upper or lower case.

Fields can be textual, numeric or contain Universally Unique Identifiers (uuids) that refer to other objects. See the section *Data types* for details.

The only operators you can use on uuid fields are EQ and NE, which can also be used on all other field types.

Other operators work on both textual and numeric fields, except LIKE and UNLIKE, which work only on textual fields and can take the wildcards * and ?. The * wildcard matches zero or more characters, while ? matches any one character.

All values are implicitly converted to lower case and comparisons are case insensitive unless the value is enclosed in quotes.

The operators BTW and OUT take two comparison values, enclosed in parentheses. These operators include the boundary values shown in brackets, so BTW (2,4) matches 2, 3 and 4, and OUT (2,4) matches any values except 2, 3 and 4. All other operators take only one value.

An object type with no relations selects all instances of the object.

Examples of the use of operators

The following example:

```
host name.like.hq*
```

selects all host objects with a name that matches hq*.

```
tape errorcount.lt.3
```

selects all tapes with an error count less than 3.

```
tape errorcount.btw.(1,3)
```

selects all tapes with an error count of 1, 2 or 3.

```
user
```

selects all users.

You can supply more than one selection criterion to a command. Individual criteria are ANDed together unless a field name is repeated: objects are returned only if they satisfy all the criteria. Multiple relations on the same field are treated as OR criteria: objects satisfying one or more of these criteria are returned. For example:

```
device errorcount.gt.0 name.like."HQ*"
```

selects all devices with an error count greater than 0 and a name that matches HQ*. Note that enclosing the name value in quotes makes the comparison case sensitive.

```
tape errorcount.lt.5 status.eq.blank status.eq.scratch
```

selects tapes with an error count less than 5 and a status of either BLANK or SCRATCH.

Note	*You cannot embed options in selection criteria.*

Embedded selection criteria

Every object has a Universally Unique Identifier (uuid). When one database object refers to another, it stores the uuid of that object. When a command expects a uuid, you can use selection criteria, but the criteria must be preceded by an @ symbol. You can use this technique to set values, or express selections that relate to other objects.

When you use embedded selection criteria within other selection criteria, you can enclose the embedded criteria in parentheses to avoid ambiguity.

Examples

To show the names of devices accessible by the host cygnus (in slash syntax) enter:

```
show /items=name device -
     refaccessiblebyhost.eq.(@host name.eq.cygnus)
```

The same command issued in dash syntax is:

```
show -item name device -
     refaccessiblebyhost.eq.(@host name.eq.cygnus)
```

To show details of the devices accessible by the host venus and with an error count greater than zero enter:

```
show device refaccessiblebyhost.eq.(@host name.eq.venus) -
            errorcount.gt.0
```

In this example, the parentheses are required to delimit the criteria, as errorcount refers to the device, not the host. Alternatively, you could express the command as follows:

```
show device errorcount.gt.0 -
            refaccessiblebyhost.eq.@host name.eq.venus
```

Tagged selection criteria

The REPORT command allows you to produce reports and format them for printing. You use *tagged* (*template*) criteria with variable values when you save a standard report whose selection criteria will be entered when the report is run. For example, if you have a standard report of users called USER5, which includes selection by user name specified when the REPORT command is issued, you could initially set up the selection criteria as follows:

```
user name.like.tag.who
```

The report could then be run by using the command:

```
report /name:user5 /with:who:a*
```

which would use runtime selection criteria equivalent to:

```
user name.like.a*
```

If a default value is required as well as a tag, the following form is used:

```
user name.like.*.tag.who
```

In this example, the default is * (all values).

Data types

Each field in an object has a data type, of which you must be aware when specifying values in selection criteria. For example, if you want to show the tapes that are due to expire within a given time, you specify a *deltatime*, which is a relative period of time. For example, +0 is the same time as the base time: if the base time is the current time, the deltatime +0 is now. The data types used by OpenMedia are as follows:

- Timestamp (absolute time)
- Deltatime (interval of time)
- Bool
- Enum

- Uuid

Use the DISPLAY command to show the data types of fields.

Timestamp

A timestamp represents an actual time. It takes the format:

yyyy-mm-dd hh:mm:ss.ffff

where
yyyy is the year
mm is the month
dd is the day
hh is hours
mm is minutes
ss is seconds
ffff is fractions of a second (generally ignored)

For example, 1997-10-13 22:15 is quarter past ten on 13th October, 1997.

You can omit parts of a timestamp. The current date and time is applied to any missing part up to the first part present. After that point, fields are defaulted to 1 for the month and day and 0 for the time fields. For example:

```
15:00          is 15:00 today
-1-2 12        is Midday on January 2nd this year
```

To enter the current date and time, enter a single space, dash or colon. The following all denote now:

```
-
:
::0
```

In the last example, the current date and time is resolved to the nearest minute, as the seconds are zero.

Deltatime

A deltatime specifies an interval of time. It takes the format:

[+|-]dddd hh:mm:ss

Where
dddd is the number of days
hh is the number of hours
mm is the number of minutes
ss is the number of seconds

Examples of deltatime fields are the onsite and offsite duration and the retention period specified in media pool objects. All these fields define periods of time. The onsite duration is the length of time that the tape is kept onsite after it has been used. The offsite duration is the length of time that the tape is kept in offsite storage. The retention is used to calculate the expiration date of a tape: the expiration date is the date and time at which the tape is moved from the scratch pool (a timestamp) plus the retention (a deltatime).

Other times are typically specified using timestamps.

Combination dates and times

When you specify a timestamp field, you can use a combination of timestamp and deltatime syntax. This enables you to express concepts such as 19:00 tomorrow, in one year's time and last January. The syntax for this is:

```
[timestamp|keyword][{+|-} deltatime
```

A leading - in the deltatime denotes a negative interval (before the timestamp) and a leading + denotes a positive interval (after the timestamp).

The following keywords are supported:

NOW
: the current date and time to second resolution

TODAY
: start of today (previous midnight)

TOMORROW
: start of tomorrow (next midnight)

YESTERDAY
: start of yesterday (midnight)

These keywords are translated relative to the timestamp. If parts of the timestamp are omitted, the missing fields are defaulted as normal. If the timestamp is omitted completely, the deltatime is applied to now.

For example:

```
0:0-1     is midnight yesterday
0:0+1     is midnight tonight
19:00+1   is 7pm tomorrow
+365      is the current date and time next year (unless a leap year)
-1        is this time yesterday
:0:0+1    is this time tomorrow (zero seconds)
:0+1      is the beginning of this hour tomorrow
```

It is best to use the numeric variants in command files if there is any possibility that the command file may be used with a non-English variant of OpenMedia. Values containing +, - or spaces must be enclosed in quotes.

Bool

Bool is a logical switch, which takes the value YES or NO.

Enum

The value of an enum data type is one of a series of keywords (you can use the DISPLAY command to show the available keywords - see the section *Using the Display Command* earlier in the chapter for details).

Uuid

A uuid is an internal reference to another object, represented by an alphanumeric string. As this data type is used to reference another object, it is usually set using embedded selection criteria.

Item lists

The following commands allow you use the ITEMS option to provide a list of values:

- SHOW
- ADD
- COPY
- SET
- REPORT
- CLOSE SEGMENT
- OPEN SAVESET

See the sections on the individual commands for syntax details.

Setting defaults

The MODE command allows you to set defaults that affect a number of CLI commands. These defaults are:

- Use of quiet or verbose mode of working. In verbose mode, progress messages are output by the CLI; in quiet mode, they are suppressed. This default can be overridden for individual commands using the QUIET and VERBOSE options.
- Setting of a limit for the object manipulating commands SET and DELETE. If selection criteria on such commands select more than the stated limit, the command is not executed. This default can be overridden for individual commands using the LIMIT option.
- Setting of the default assignment style to use the name of an object or its uuid in response to SHOW and REPORT commands. You can also set the default to show both or neither of these items.

To set a default, issue the MODE command, followed by the keyword identifying the default and a value if appropriate. To see the current defaults, issue the mode command without a keyword.

Examples

`mode quiet`

selects quiet working for all CLI commands

`mode verbose`

selects verbose working for all CLI commands

`mode assign uuid`

shows the object uuid in reference fields returned by SHOW and REPORT

`mode assign name`

shows the object name in reference fields returned by SHOW and REPORT

`mode assign both`

shows both the object name and uuid in reference fields returned by SHOW and REPORT

mode assign none

shows neither the object name nor uuid in reference fields returned by SHOW and REPORT

mode limit 10

sets the limit for database manipulation commands to 10. Any command whose selection criteria return more than 10 objects will be ignored.

mode nolimit

removes any limit set for database manipulation commands.

Warning	*You are strongly advised to set a realistic limit using the limit parameter, as a safeguard against accidental deletion of an unlimited number of objects. You can override this limit as required by specifying a limit option on individual commands.*

CLI commands

The following sections list the CLI commands in alphabetical order. Each section describes a command, describes the syntax and lists the options available with the command.

ADD

add *object_type*

Adds an object to the database, with fields set to values specified via the ITEMS or FILE options. The options available with ADD are:

/FILE = *filename*
-file *filename*

Directs input from the specified file. Cannot be used with ITEMS.

/HEADER
-header

Should only be used when reloading data from a file.

/ITEMS = (*field1*[, *field2* ...])items
-item *field1* [-item *field2*...]

Sets the values for specified fields. Cannot be used with FILE.

/QUIET
-quiet

Sets quiet mode for the command.

/RETRY = *maxretries*
-retry *maxretries*

Allows multiple passes through a file of object definitions. This is useful if you want to eliminate forward referencing problems when reloading self referencing objects into the database.

/UPGRADE
-upgrade

Can be used when upgrading the database. Allows invalid items to be ignored rather than causing an error.

/VERBOSE
-verbose

Sets verbose mode for the command.

Examples

```
add host /items=(name=gannet,comment="Paul's PC"
add host -item name:gannet -item comment:"Paul's PC"
```

ASSIGN

assign *main_object criteria1* [*criteria2*...] -
***referencing_field* -**
***referenced_object criteria1* [*criteria2*...]**

Defines references between objects. The options available with ASSIGN are:

/LIMIT = *maxrecords*
-limit *maxrecords*

Assigns a limit for the maximum matches on the main object.

/QUIET
-quiet

Sets quiet mode for the command.

/RANK = *rank*
-rank *rank*

Used to order relations that have an associated rank.

/SORT = *sortkey*
-sort *sortkey*

Provides an object field or key to be used as a sort key on the main object for items to be returned.

/VERBOSE
-verbose

Sets verbose mode for the command.

Example

```
assign device name.eq.DLT5 -
       refaccessiblebyhost host name.eq.cygnus
```

See later in the chapter for examples of the use of this command.

CLEAN DEVICE

clean device [*criteria1*...]

Cleans a device. The device may already reserved in a context, or a device that is not reserved, in which case it is specified by selection criteria. You can clean a number of devices by specifying selection criteria and a LIMIT option. The options available with CLEAN DEVICE are:

/CONTEXT = *n*
-context *n*

Specifies the context.

/LIMIT = *maxrecords*
-limit *maxrecords*

Sets a limit for the maximum number of matches.

/QUIET
-quiet

Sets quiet mode for the command.

/VERBOSE
-verbose

Sets verbose mode for the command.

Example

```
clean device name.eq.saturn-1
```

CLOSE SAVESET

close saveset

Closes a saveset that has been opened by an OPEN SAVESET command. The options available with CLOSE SAVESET are:

/CONTEXT = *n*
-context *n*

Specifies the context.

/DISCARD
-discard

Discards all information accumulated in the saveset.

CLOSE SEGMENT

close segment

Closes a segment that has been opened by an OPEN SEGMENT command, including size, absolute tape position and relative tape position information supplied via ITEMS or FILE options. The options available with CLOSE SEGMENT are:

/CONTEXT = *n*
-context *n*

Specifies the context.

/FILE = *filename*
-file *filename*

Directs input from the specified file.

/ITEMS = [*field1*=*value1*[, *field2*=*value2*...]items
-item *field1*:*value1* [-item *field2*:*value2*...]

Specifies one or more of the segment fields: SizeKb, ClientAbsoluteTapePosition, ClientRelativeTapePosition.

/STATUS = [UNTERMINATED | NORMAL | FULL]
-status [unterminated | normal | full]

Sets the status of the segment. If this option is not specified, normal is assumed.

/STREAM = *n*
-stream *n*

Specifies the stream number.

/TAPECONTEXT = *n*
-tapecontext *n*

Used to specify a tape in another context.

COPY

copy *object_type* [*criteria1...*]

Adds one or more objects to the database by copying one or more objects and modifying certain fields specified via FILE or ITEMS options. If the criteria select more than one object, the ITEMS are supplied to each of them. A single object may also be replicated.

The options available with COPY are:

/FILE = *filename*
-file *filename*

Directs input from the specified file. Cannot be used with ITEMS.

/ITEMS = (*field1*=*value1*[:*highvalue1*],
** *field2*=*value2*[:*highvalue2*] ...)**
-item *field1*:*value1*[:*highvalue1*] -
** [-item *field2*:*value2*[:*highvalue2*]...]**

Sets new values for specified fields. You can optionally specify a high value on one or more items, in which case multiple objects are added. After adding each object, the field value is incremented until the high value is reached or the specified LIMIT is reached. Cannot be used with FILE.

/LIMIT = *maximumadded*
-limit *maximumadded*

Specifies the maximum number of objects to be added.

/QUIET
-quiet

Sets quiet mode for the command.

/VERBOSE
-verbose

Sets verbose mode for the command.

Examples

```
copy useraccount host.eq.mypc nativeuser.eq.fred -
                /items=(nativeuser=john)
copy useraccount host.eq.mypc nativeuser.eq.fred -
                -item nativeuser:john
copy tape external.eq.zz_template -
          /items=(external:"ZZ0001":"ZZ0099", -
                  internal:"ZZ0001":"ZZ0099")
copy tape external.eq.zz_template -
          -item external:"ZZ0001":"ZZ0099" -
          -item internal:"ZZ0001":"ZZ0099"
```

```
copy useraccount host.eq.moon /items=(host=stars)
copy useraccount host.eq.moon -item host:stars
```

DEASSIGN

deassign *main_object criteria1* [*criteria2...*] -
***referencing_field* -**
***referenced_object criteria1* [*criteria2...*]**

Removes references between database objects; the opposite of ASSIGN. The options available with DEASSIGN are:

/LIMIT = *maxrecords*
-limit *maxrecords*

Assigns a limit for the maximum matches on the main object and the total number of intersections.

/QUIET
-quiet

Sets quiet mode for the command.

/SORT = *sortkey*
-sort *sortkey*

Provides an object field or key to be used as a sort key on the main object for items to be returned.

/VERBOSE
-verbose

Sets verbose mode for the command.

Example

```
deassign device name.eq.DLT5 -
         refaccessiblebyhost host name.eq.cygnus
```

DELETE

delete *object_type criteria1* [*criteria2...*]

Deletes one or more database objects specified by selection criteria. The options available with DELETE are:

/CONFIRM
-CONFIRM

Requests confirmation before deleting each object. To continue with the deletion following a request for confirmation, enter a response beginning with T, t, Y or y.

/LIMIT = *maxrecords*
-limit *maxrecords*

Assigns a limit for the maximum matches allowed. If it is exceeded, the command is not executed. A zero limit is treated as no limit.

/QUIET
-quiet

Sets quiet mode for the command.

/SORT = *sortkey*
-sort *sortkey*

Provides an object field or key to be used as a sort key for the items to be returned. This option is only useful when combined with a VERBOSE or CONFIRM option.

/VERBOSE
-verbose

Sets verbose mode for the command.

Examples

```
delete user name.like.test* /confirm
delete user name.like.test* -confirm
delete host name.like.hq* /limit=0
delete host name.like.hq* -limit 0
```

DEREGISTER

deregister {prompts | changes}

Deregisters from tracking prompts or changes: undoes a previous REGISTER PROMPTS or REGISTER CHANGES command.

DISPLAY

display [*object_type*]

Displays details of the specified object type. This command displays the structure of the object in terms of the field names and keys within it, and not the values set in a particular object. The data type of each field is also displayed. See the section on *Data types* for further information.

Any fields with the annotation ND are not held in the database, and cannot be accessed via the CLI.

If you specify the object type as * (asterisk) all object types are displayed.

If you do not specify an object type, the response to the command shows the names of all object types.

The options available with DISPLAY are:

/FILE = *outputfile*
-file *outputfile*

Directs output to the specified file.

/HEADER
-header

Displays fields in the object header.

ECHO

echo {on | off}

Turns echoing command line on or off. This only has an effect if the command is input from a file (using input redirection or piping). By default, echoing is turned off until it is set on.

END

end

Exits from the CLI and terminates the background process, which holds connection information, tape and device reservations and open saveset information.

To exit from the CLI without terminating the background process, use EXIT or QUIT.

EXIT

exit

This commands is synonymous with QUIT, causing the CLI to exit, without terminating the background process.

EXPIRE SAVESET

expire saveset [*criteria1...*]

Expires selected savesets. Once a saveset is expired, it can be removed using the REMOVE SAVESET command.

The options available with EXPIRE SAVESET are:

/LIMIT = *maxrecords*
-limit *maxrecords*

Sets a limit for the maximum number of matches.

/QUIET
-quiet

Sets quiet mode for the command.

/VERBOSE
-verbose

Sets verbose mode for the command.

HELP

help [command verb]

Gives help on all commands, or a specific command in much more detail. Help is given in either the slash or dash form, depending on which syntax was last used. Use /MODE to set the style to slash, or -mode to set the style to dash.

The option available with HELP is:

/MODE
-mode

Sets the mode of the help to slash or dash syntax.

LOGIN

login [*username*] [*password*]

Establishes a connection to the server and establishes your access rights.

You do not need to specify a username and password if the OpenMedia object database contains a user account object that matches the name of the host system and the username under which you are logged into the host to a valid OpenMedia user. The CLI automatically logs you in, using native accreditation.

If you specify a username, it must be the name of an OpenMedia user. The user's password, as held in the object database, is also required. If it is not supplied at the login command, it is requested.

You can specify the locality using the NEIGHBORHOOD option. If you do not specify this option, the default used is the neighborhood locality held in the local configuration file LOCAL_CONFIG.DAT. To see a list of the available localities, use the LIST option.

To attach directly to the database, use the DATABASE option. Native accreditation is not allowed when connecting directly to the database: you must specify the name and password of a user with MANAGER privilege. However, you do not require a username or password if the database is empty.

The options available with the LOGIN command are:

/ALL
-all

When used in conjunction with the LIST option, this causes all neighborhoods to be displayed, not just those available to this CLI.

/DATABASE
-database

Used to connect the CLI directly to the database without using the normal server, when setting up an empty database.

/LIST
-list

Lists available neighborhood localities without attempting a login.

/NEIGHBORHOOD = *neighborhoodname*
-neighborhood *neighborhoodname*

Names the neighborhood locality to which the client is to be attached.

Note	*You do not need to issued a login command when connecting to the default locality using native accreditation. The first command you issue logs you in automatically using these defaults.*

MODE

mode [quiet | verbose | nolimit | limit *value* |
assign *style*]

Sets defaults, or displays the current settings if no keyword is given.

Style is one of: NONE, NAME, UUID or BOTH

You can use this command to:

- Specify the default limit for commands that manipulate database objects
- Toggle between quiet and verbose output of progress messages

- Indicate whether commands such as SHOW should return the name field, uuid, neither or both whenever a uuid occurs in an object

These defaults remain in force until the next MODE command or the next END command, at which point they revert to quiet output, a limit of 1 and assignment style of name.

To display the current default settings, issue the MODE command without a keyword.

See the section *Setting defaults* earlier in the chapter for examples of the use of the MODE command.

MONITOR

monitor

Waits for any prompts or changes to be reported, then displays the data structure. To use this command, you must first have registered for prompts or changes using REGISTER PROMPTS or REGISTER CHANGES. The MONITOR command does not terminate in the usual way. If you use the operating system interrupt (normally CTRL+C), monitoring stops when the next command is typed. On Windows systems, you may need to enter the CTRL+C twice.

MOVE

move tape *criteria1* [*criteria2*...]

The MOVE command can be used to mark one or more tapes as onsite or offsite, record the movement of tapes to a different media location, or move tapes to a new media pool. The options available with MOVE are:

/LIMIT = *maxrecords*
-limit *maxrecords*

Sets the maximum limit for the number of matches.

/LOCATION = *newmedialocation*
-location *newmedialocation*

Specifies the new media location for the tape.

/OFFSITE
-offsite

Indicates that the tape has been moved to its offsite media location.

/ONSITE
-onsite

Indicates that the tape has been moved to its onsite media location.

/POOL = *newpool*
-pool *newpool*

Specifies the new media pool for the tape, if required.

/QUIET
-quiet

Sets quiet mode for the command.

/VERBOSE
-verbose

Sets verbose mode for the command.

Examples

```
move tape externallabel.eq.pay057 /offsite
move tape externallabel.eq.pay057 -offsite
move tape internallabel.like.arc* /pool=archive1997
move tape internallable.like.arc* -pool archive1997
move tape comment.eq."Ready to ship" /location=datastore
move tape comment.eq."Ready to ship" -location datastore
```

NOTE DEVICE

note device [*criteria1...*]

Notes that one or more devices have been cleaned or serviced. (A service is assumed to include a clean.) The options available with the NOTE DEVICE command are:

/CLEANED
-cleaned

Notes that the device has been cleaned.

/LIMIT = *maxrecords*
-limit *maxrecords*

Sets a limit for the maximum number of matches.

/QUIET
-quiet

Sets quiet mode for the command.

/SERVICED
-serviced

Notes that the device has been serviced.

/VERBOSE
-verbose

Sets verbose mode for the command.

NOTE TAPE

note tape

Notes that the reserved tape associated with a given context has been initialized to the specified label, format and media type. The options available with NOTE TAPE are:

/CONTEXT = *n*
-context *n*

Specifies the context.

/FORMAT = *newformat*
-format *newformat*

Specifies the new format for the tape. If omitted, the existing value is preserved.

/INITIALIZED
-initialized

Notes that the tape has been initialized.

/INTERNALLABEL = *newlabel*
-internallabel *newlabel*

Specifies the new label for the tape. If omitted, the existing value is preserved.

/MEDIATYPE = *newmediatype*
-mediatype *newmediatype*

Specifies the new media type for the tape. If omitted, the existing value is preserved.

OPEN SAVESET

open saveset

Creates and opens a saveset information stream in the specified context. You can set fields in the saveset using ITEMS or FILE options. The options available with OPEN SAVESET are:

/CONTEXT = *n*
-context *n*

Specifies the context.

/FILE = *filename*
-file *filename*

Directs input from the specified file. Cannot be used with ITEMS.

/ITEMS = [*field1*=*value1*, *field2*=*value2*...]
-item *field1*:*value1* [-item *field2*:*value2*...]

Specifies field values in the saveset, Cannot be used with FILE.

OPEN SEGMENT

open segment

Creates and opens a saveset segment for the tape reserved in the specified context or alternative context. The options available with OPEN SEGMENT are:

/CONTEXT = *n*
-context *n*

Specifies the context.

/STREAM = *n*
-stream *n*

Specifies an arbitrary stream number, used to make the segment uniquely identifiable.

/TAPECONTEXT = *n*
-tapecontext *n*

Specifies the context that has the reserved tape.

Note	*Segment information is specified in the CLOSE SEGMENT command.*

QUIT

quit

This command is synonymous with EXIT, causing the CLI to exit, without terminating the background process.

RECONCILE JUKEBOX

reconcile jukebox *jukeboxname*

This command is reserved for jukebox integration scripts, and should not be used in any other context. For further help, see *Integrating with robot managers* or *Integrating with SCH:Robot* in chapter 2, or the documentation supplied with your robot manager.

REGISTER CHANGES

register changes

Registers for change notification. Any changes to object data in the server are reported, and displayed if a MONITOR command is active.

REGISTER PROMPTS

register prompts

Registers for prompt notification. Any outstanding prompts are tracked, and can be displayed using the STATUS PROMPTS command.

RELEASE

release {tape | device}

Releases a tape or device reserved in the specified context. The device or tape error count may be set if desired. The options available with RELEASE are:

/CONTEXT = *n*
-context *n*

Specifies the context.

/ERRORCOUNT = *n*
-errorcount *n*

Specifies a new value for the device or tape error count.

REMOVE SAVESET

remove saveset [*criteria1*...]

Removes selected savesets, which must have already expired. The options available with REMOVE SAVESET are:

/LIMIT = *maxrecords*
-limit *maxrecords*

Sets a limit for the maximum number of matches.

/QUIET
-quiet

Sets quiet mode for the command.

/VERBOSE
-verbose

Sets verbose mode for the command.

REPLY

reply [*status*] [*parameter1...*]

Sends a reply to an outstanding prompt. Use STATUS PROMPTS to see a list of outstanding prompts and identification numbers. The optional status code defaults to SUCCESS, and is only relevant for some prompts. Use the HELP REPLY command for further information. The options available with REPLY are:

/STATUS = *messagecode*
-status *messagecode*

Specifies a message code, where appropriate. The default is SUCCESS.

/TO = *promptnumber*
-to *promptnumber*

Specifies the prompt number.

REPORT

report [*object_type*] [*criteria1...*]

Runs or defines a report.

OpenMedia supplies a number of standard reports, which define sets of selection criteria and fields to display. Any selection criteria that you specify with the command are ANDed with those already in the standard report. If you specify display fields using the ITEMS option, they replace the fields defined in the standard report. You can find the standard reports in the add-browses.txt file, which is placed in the bin subdirectory of the OpenMedia top level directory at installation of a server host.

Warning *Do not amend or remove any of the standard reports.*

Use the NAME option to specify the report you want to run. If the NAME option is missing, a report is defined and run. Use the SAVE option to save a report.

If you save a report with the name set to "Site-*identifier*", where *identifier* is any combination of characters, you can view and run it from the GUI (*see Running site defined reports* in Chapter 8 for details). Note that the use of quotes in the name definition makes it case-sensitive. The GUI displays these reports by title and comment. Use the TITLE option to set the title, and the SET command to add a comment.

When you run a site defined report containing tagged selection criteria, the GUI automatically creates and displays a selection window for entry of the appropriate runtime criteria. See the section on *Tagged selection criteria* earlier in this chapter for more information about tagged selection criteria.

Reports are stored in report, criteria, criteria-relation, report-column tables. The add-browses command script allows you to remove an entire report from the OpenMedia database. This command script is placed in the bin subdirectory of the OpenMedia top level directory during installation of a server host.

For example, to delete a report named Site-rep01, exit the CLI, move to the OpenMedia bin subdirectory and enter:

add-browses clear Site-rep01

from the command line.

The options available with REPORT are:

/EXPORT
-export

Eliminates column alignment and suppresses the column header (unless HEADER is also specified).

/FILE = *filename*
-file *filename*

Directs output to the specified file.

/HEADER
-header

Prints the column heading line, if EXPORT is used.

/ITEMS = (*formatspec1*[, *formatspec2* ...])
-item *formatspec1* [-item *formatspec2*...]

Specifies the alignment, width and title of columns, where *formatspec* specifies the column in the report that shows the field, as follows:

***fieldname* [:[L | R]*width*[:*title*]].**

/LIMIT = *maxrecords*
-limit *maxrecords*

Sets a maximum limit for the number of matches.

/LIST
-list

Lists the template selection tags.

/MAXIMUM = *maxwidth*
-maximum *maxwidth*

Applies a maximum column width.

/NAME = *reportname*
-name *reportname*

Specifies the name of the report.

/PAGESIZE = *linesperpage*
-pagesize *linesperpage*

Specifies the page size if pagination is required.

/SAVE
-save

Saves the report in the database for later use.

/SEPARATOR = *columnseparatorstring*
-separator *columnseparatorstring*

Defines the column separators in the output file. Can be a character or string of characters. Defaults to a space, or vertical bar if EXPORT is used.

/SORT = *sortkey*
-sort *sortkey*

Provides an object field or key to be used as a sort key for the items to be returned.

/TITLE = *reporttitle*
-title *reporttitle*

Specifies the title printed on the report.

/WITH = (*tag1*=*value1*[:*highvalue1*]-
[, *tag2*=*value2*[:*highvalue2*]...])
-with *tag1*:*value1*[:*highvalue1*] -
[-with *tag2*:*value2*[:*highvalue2*]...]

Provides values for any tagged criteria in a saved report.

Examples

```
report /name=badtapes onsite.eq.0 -
       /item=(internallabel:L10:"Label", -
              externallabel:L10:"External")
report -name badtapes onsite.eq.0 -
       -item internallabel:L10:"Label" -
       -item externallabel:L10:"External"
report /name=user3 /with:name:hq*
report -name user3 -with name:hq*
```

REQUEST

request message [*object_type* [*criteria1*...]]

Raises a message prompt and awaits a response. If the message is about a specific object (typically a device or jukebox), this can be defined.

Examples

```
request "Are any operators in the machine room?"
request "Is the DLT OK?" device name.eq.hq_dlt4000
```

RESERVE

reserve tape *criteria1* [*criteria2*...]
[device *criteriaa* [*criteriab*]]

Reserves a tape matching the specified selection criteria, together with a suitable device, and loads the tape on that device. If a device is specified, the device must match the specification. If a device is already reserved in the context from a previous RESERVE TAPE, it is reused. The tape is reserved for read-only or write access as appropriate. The options available with RESERVE are:

/CHECK
-check

Simulates please-check-label callback functionality.

/CONTEXT = *n*
-context *n*

Specifies the context.

/FILE = *filename*
-file *filename*

Directs output to the specified file.

/NOTEONLY
-noteonly

Ensures that no tape loading is performed.

/READ
-read

Reserves the tape for read-only access.

/SCRIPT = *filename*
-script *filename*

Specifies a shell script that directs input from a file.

/WRITE
-write

Reserves the tape for write access.

SCRATCH

scratch tape [*criteria1...*]

Scratches one or more expired tapes. If the tape reserved in a context is to be scratched, /CONTEXT may be specified instead of selection criteria. The options available with SCRATCH are:

/CONTEXT = *n*
-context *n*

Specifies the context.

/LIMIT = *maxrecords*
-limit *maxrecords*

Sets a maximum limit for the number of matches.

/QUIET
-quiet

Sets quiet mode for the command.

/SORT = *sortkey*
-sort *sortkey*

Specifies an object field or key to be used as a sort key for the items returned.

/VERBOSE
-verbose

Sets verbose mode for the command.

SERVER DATABASE

server database {deaccess | reaccess}

Requests that the server deaccess or reaccess its underlying database. This command may be used by procedures to backup the database, and must be used with care. The message code specified in the STATUS option is returned to any requests that need access to the database while it is deaccessed.

The option available with SERVER DATABASE is:

/STATUS = *messagecode*
-status *messagecode*

Used only with the deaccess keyword, and defines the status code which will be returned to any request requiring the database while it is deaccessed.

SERVER SHUTDOWN

server shutdown

Shuts down the server. The server is shut down cleanly by default. The option available with SERVER SHUTDOWN is:

/EMERGENCY
-emergency

Performs an emergency shutdown, as opposed to the default clean shutdown.

SERVER TRACE

server trace [*tracelevel*]

Sets the trace level of the server, and/or gets the server to start a new trace file. For use only as directed by technical support. The option available with SERVER TRACE is:

/NEWFILE
-newfile

Indicates that a new trace file is started.

SET

set *object_type* [*criteria1...*]

Modifies one or more database objects, changing the values of fields specified in ITEMS or FILE options. The options available with SET are:

/ASSIGNS
-assigns

Includes many-to-many references to other database objects. Without this option, such references are ignored.

/CONFIRM
-confirm

Requests confirmation before modifying each object.

/FILE = *filename*
-file *filename*

Directs input from the specified file. Cannot be used with ITEMS.

/ITEMS = (*field1*=*value1*[, *field2*=*value2* ...])
-item *field1*:*value1* [-item *field2*:*value2*...]

Specifies the values of the fields to be changed. Cannot be used with FILE.

/LIMIT = *maxrecords*
-limit *maxrecords*

Assigns a limit for the maximum matches allowed. If this limit is exceeded, the command is not executed.

/QUIET
-quiet

Sets quiet mode for the command.

/SORT = *sortkey*
-sort *sortkey*

Provides a an object field or key to be used as a sort key for the items to be returned. This option is only useful when combined with VERBOSE or CONFIRM.

/VERBOSE
-verbose

Sets verbose mode for the command.

Examples

```
set user name.eq.jdoe /item=(mailaddr:"jdoe@eagle.com")
set user name.eq.jdoe -item mailaddr:"jdoe@eagle.com"
```

SHOW

show *object_type criteria1* [*criteria2*...]

Shows one or more database objects. You can use ITEMS to specify the fields to be displayed. If you do not specify ITEMS, all simple fields are displayed. Related objects are shown if you specify ASSIGNS, and object header information is displayed if you specify HEADER.

If you specify NAME without ITEMS, only the names of matching objects are shown.

The style of assignments output in response to this command is determined by the default mode. Use the MODE command to determine the current setting, and MODE ASSIGN STYLE to change it.

The options available with SHOW are:

/ASSIGNS
-assigns

Includes many-to-many references to other database objects. Without this option, such references are omitted.

/ENVIRONMENT
-environment

Produces information about the current environment in which the CLI is operating, for example:

```
HostName = jupiter.mycompany.com
HostSimpleName = jupiter
HostRealName = jupiter.mycompany.com
HostAddress = 193.129.117.101
OSName = Windows NT
OSVersion = 3.51
Accredited = NO
ServerNeighborhood = London_Data_Center
ProductName = HYPERtape OpenMedia Manager
ProductVersion = 3.1 Level 10
ProductDate = 1998-03-14
ProductBuild = 12-010
ProductProducer = MultiStream Systems
Date = 1998-03-14
Time = 16:17:44
```

The object, selection criteria and options SORT, LIMIT, ASSIGNS and HEADER are ignored if specified with ENVIRONMENT.

/FILE = *filename*
-file *filename*

Directs output to the specified file.

/HEADER
-header

Includes the headers of returned objects (which includes the uuid).

/ITEMS = (*field1* [, *field2* ...])
-item *field1* [-item *field2*...]

Limits the output to specified fields.

/LIMIT = *maxrecords*
-limit *maxrecords*

Assigns a limit for the maximum matches allowed. If this limit is exceeded, the command is not executed.

/NAME
-name

Limits the output to the name of the object, if there is one. This is equivalent to specifying the name field in the ITEMS option. In tape objects, the name field is the external label. Some objects, for example segments, do not have a name field.

/QUIET
-quiet

Sets quiet mode for the command.

/RAW
-raw

May only be used with a single item, which it displays without any annotation. This is particularly useful in UNIX scripts.

/SCRIPT = *filename*
-script *filename*

Specifies a shell script that directs output to a file.

/SORT = *sortkey*
-sort *sortkey*

Provides an object field or key to be used as a sort key for the items to be returned.

/VERBOSE
-verbose

Sets verbose mode for the command. This can be useful if using the FILE option.

Examples

```
show host name.like.hq* /limit=0
show host name.like.hq* -limit 0
```

STATUS

status {tape | device | context | prompts}

STATUS CONTEXT shows information held in the current context, including a reserved tape, reserved device and an open saveset.

STATUS TAPE and STATUS DEVICE display the tape status or device status for the reserved tape or reserved device in the specified context.

STATUS PROMPTS lists the outstanding prompts. Prompt registration must be in effect from an earlier REGISTER PROMPTS command.

The options available with STATUS are:

/ALL
-all

Use this option to show information in all active contexts (not used with STATUS PROMPTS).

/CONTEXT = *n*
-context *n*

Specifies the context (not used with STATUS PROMPTS).

/FILE = *filename*
-file *filename*

Directs output to the specified file.

/SCRIPT = *filename*
-script *filename*

Specifies a shell script to direct output to a file.

UNLOAD

unload [device *criteria1* [*criteria2*...]]

Unloads a tape which has been loaded in the specified context, or unloads the device specified in the selection criteria. The options available with UNLOAD are:

/CONTEXT = *n*
context *n*

Specifies the context.

/NOTEONLY
-noteonly

Ensures no tape unloading is actually performed - used to inform the server that the device has been unloaded.

/STATUS = *messagecode*
-status *messagecode*

Specifies the message code for the unload, if any status other than SUCCESS should be recorded.

VERSION

version

Shows information about the version of various software components. If connected to a server, the current neighborhood locality name and user privileges are also shown. If verbose mode is set, server statistics are shown.

Using ASSIGN and DEASSIGN commands

Assign and deassign commands are used to manipulate the relationships between objects, for example the media types that a device can read, or the hosts that can access a device.

When an object can relate to only one other object at a time, for example the locality to which a host relates, ASSIGN replaces the existing link with another and DEASSIGN clears it, if permitted.

When an object can relate to more than one other object at a time, for example the media types that a device can read, ASSIGN adds one or more new references and DEASSIGN removes one or more references.

Examples

In the following examples, *xxx* represents selection criteria.

```
ASSIGN device xxx refpreferredbylocation location xxx
ASSIGN device xxx refwritemediatypes mediatype xxx
ASSIGN medialocation xxx refpreferredbydevice -
       device xxx /rank=n
```

Note	*Although you can create assignments using ASSIGN, SET and ADD, breaking such relationships should always be accomplished using DEASSIGN.*

Writing scripts

Take care to note the following when you invoke CLI commands from a batch file or shell script. Some of the underlying problems can be eliminated by piping CLI commands into the CLI, as in:

```
mmcli < various-commands.txt
```

Windows NT batch files

When a CLI command line requires double quotes, these must be tripled to avoid the batch interpreter stripping them out. For example:

```
mmcli set user name.eq.tom -
          /comment:"""Tom Smith, Finance"""
```

Alternatively, use the single quote character:

```
mmcli set user name.eq.tom -
          /comment:'Tom Smith, Finance'
```

UNIX shell scripts

When a CLI command line requires double quotes or other shell meta-characters, these must be escaped using the backslash character or quoted as appropriate. For example:

```
mmcli set user name.eq.tom -item -
          comment:\"Tom Smith, Finance\"
mmcli show host name.like."a*"
```

Note that in the second of these examples, the double quotes are absorbed by the shell, so the system is presented with name.like.a*.

Using SHOW and SET commands

The following examples illustrate the uses of the SHOW and SET commands:

Example 1: Updating a media type using an intermediate file

```
M:\ mmcli show mediatype name.eq.dat /file:dat.tmp
M:\ type dat.tmp
Name = dat
Comment = "4mm DAT tapes, 90m, 4Gb with compaction"
NameOfLocality = "HQ"
CapacityKb = 4000000
NameofPhysicalType = "4mm"
M:\ perl munge.pl mediatype dat.tmp
M:\ mmcli set mediatype /file:dat.tmp
<end>
```

Example 2: Adding a new host

```
M:\ type host.tmp
Name = phobos
Comment = "Jennifer's PDA"
UniqueName = "phobos.whizz.com"
M:\ mmcli add host /file:host.tmp
```

Example 3: Using a sort specification

```
M:\ mmcli show tape internallabel.like.YZ* /sort=barcode
```

Example 4: Updating a media type using the command line

```
M:\ mmcli set mediatype name.eq.dat /items:(Comment:"90m DAT")
```

Example 5: Associating a writeable media type with a device

```
M:\ mmcli assign device name.eq.dlt_1 refwritemediatypes mediatype name.eq.dat
```

Example 6: File format using the ASSIGNS option

```
M:\ mmcli show mediatype name.eq.dat /file:dat.tmp /assigns /header
M:\ type dat.tmp
DH_Uuid = c01efd91c1ea.80b0.02.c1.81.75.65.ff.ff.00
DH_ModCount = 4
DH_Deleted = NO
Name = dat
Comment = "4mm DAT tapes, 90m, 4Gb with compaction"
NameOfLocality = "HQ"
CapacityKb = 4000000
NameOfPhysicalType = "4m"
RefReadDevices(1) = "alpha_1"
RefReadDevices(2) = "beta_2"
RefWriteDevices(1) = "alpha_1"
<end>
```

Example 7: File format with selected fields using the ASSIGNS option

```
M:\ mmcli show mediatype name.eq.dat /file:dat.tmp /assigns /items:(name,refreaddevices)
M:\ type dat.tmp
Name = dat
RefReadDevices(1) = "alpha_1"
RefReadDevices(2) = "beta_2"
<end>
```

Example 8: Adding an object using Uuid references

```
M:\ type useracc.tmp
Host = phobos
NativeUser = John.Doe
UuidOfUser = @user name.eq.J_Doe
UuidOfDomain = @locality name.eq.Headquarters
M:\ mmcli add useraccount /file:useracc.tmp
```

5

Developing a storage management strategy

Storage management requirements

The aim of a storage management strategy is to match your offline storage resources to your users' requirements, as efficiently and effectively as possible. It will balance the demand for tapes and offline facilities against the requirements of the enterprise as a whole, the limitations under which you work and the facilities and resources available to you.

Offline storage demands arise for purposes such as:

- Backups
- Archiving
- Data for transport elsewhere

Selecting the most appropriate tape and device to use for an individual job can be complicated. For example, some jobs may require a blank volume or set of volumes while for other jobs it is more efficient to append data to a volume that already contains related data.

Using OpenMedia, you can organize your resources so that when a user or application (such as a backup program) makes a request for offline storage, the system selects the most appropriate device and tape in the circumstances. This selection will be determined by a number of factors.

These may include:

- Matching the media types of tapes and devices to ensure compatibility
- Determining whether data needs to be written to a blank volume or appended to other data that may be related in some way
- How long the data is to be retained
- Whether data is to be sent offsite, and if so, where it is to go
- Saving time, for example by using a tape that is already loaded, or choosing the tape that is closest to the device to be used

OpenMedia performs some checks automatically, but it also allows you to build factors into the OpenMedia database that describe your enterprise and help you to monitor its progress and control how it works.

Device and media management

Jobs such as regular backups that are initiated from a central control point can be automated and scheduled, but user initiated saves are more difficult to handle. The key to handling such requests lies in managing devices and media efficiently.

The OpenMedia database relates tapes to media locations, media types and tape pools. Similarly, devices in the database are related to media types and jukeboxes. You can set preferences, for example you can identify the devices closest to the tape store and make sure that they are selected before others. When a user requests a tape, OpenMedia can identify the tapes available from the appropriate tape pool, select the tape and device to be used, and issue a prompt to an operator or obotic jukebox to load the tape on the device.

There are also a number of standard database interrogations that you can display via the GUI and produce reports. Using these interrogations, you can monitor tape usage and error counts, identify tapes that are due to go offsite or return onsite, and generally keep track of how the tapes are being used.

Once a tape has been cleaned or moved, this information is stored in the database, so that you can more easily control your tape cleaning, shipment and recycling schedules.

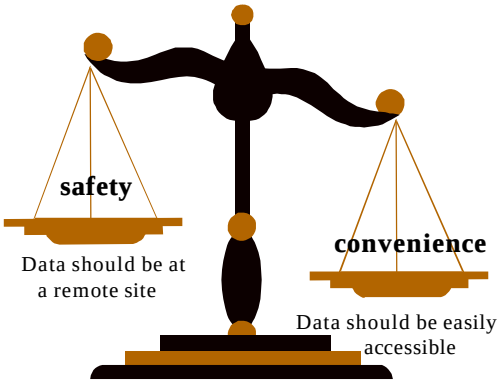
Retention of data

Data that is written to tape has a limited life. In principle, backup data is of no further use once a more recent version of the file system exists, and documents that are archived for legal purposes are no longer required at the end of the statutory term. Generally, backup data has a shorter lifespan than archive data.

OpenMedia has facilities for identifying and removing expired savesets, and automatically calculates an overall expiration date for the data on a tape so that you can easily identify tapes that no longer contain current data. You can then use the GUI or CLI to scratch the tapes and return them to a scratch media pool ready for reuse.

Storing tapes offsite

Most sites have conflicting requirements for storing offline data. On one hand there may be a requirement for easy access, for example so that you can restore accidentally deleted files quickly. On the other hand, there may be a need to hold the data at a remote site to prevent its loss in a major disaster.



The offsite paradox

The resolution of these conflicting requirements varies from site to site. In general, a site settles on a system that results in least exposure to risk coupled with highest availability. For example, a typical site may keep backups onsite for 24 hours before transferring them to an offsite store.

In this way, backups of the system to a point 24 hours prior to any disaster are secured at a remote site, but recent backups are easily available to restore individual files that have been accidentally deleted. Full backups made at weekends may be moved offsite immediately. In all cases, the tapes are returned to the site before they are required for reuse.

Developing a backup strategy

Backup management is a large part of the storage management of any enterprise. In particular, it is the system manager's job to formulate a backup strategy that keeps data safe from destruction, while allowing users to access it on demand.

In an ideal world, a copy of all information held on a computer system would also exist in an easily accessible form in a separate location, so that instantaneous recovery would be possible in the event of a mishap such as accidental file deletion, disk failure, or destruction of systems.

In practice, of course, this ideal is often not possible. The backup strategy you employ attempts to provide the best approximation to the ideal model, given the limitations of resources available. It also needs to take account of the recovery method and the recovery order in the event of a catastrophic loss of data.

A further possible requirement of a backup strategy is to allow you to recreate earlier instances of an environment so that you can research problems that arise when, for example, data is damaged but the error is not discovered until much later.

In effect, a typical backup strategy involves choosing when to take full and incremental backups, how long to keep them, and how soon to move them offsite for protection from disaster.

Limitations affecting a backup strategy

The limitations under which you work and which make the ideal of instantaneous recovery from any mishap impossible include the:

- Time available for backups
- Devices available for backing up
- Quantity of data to be backed up
- Cost and quantity of storage media required to hold backups

Time available for backups

It is obviously not practical to perform backups every minute of the day, especially when you need exclusive access to the data you are backing up. It is therefore normal to reserve a period of time during every day for making backups. Conventionally, this slot of time (the *backup window*) is scheduled when system usage by both users and applications is minimal, thus maximizing the resources available for backup and minimizing the inconvenience to users.

It is good practice to keep the backup window as short as possible, as it is often a time during which applications should not run. You may have to trade off the comprehensiveness of the backup against the time it takes, to reduce the backup window. This will affect the types of backup you employ.

Devices available for backing up

The types and numbers of devices available determine the amount of data you can back up in each backup window, and may affect the way you organize your backups.

Backups are usually made to tape or cartridge devices, because they are removable, relatively robust and offer low storage cost compared to disks. These devices range from low capacity media such as 9 track tape and TK50 cartridge to high capacity DLT and 8mm technologies. Higher capacities of storage can be made automatically available by the use of stacker and robotic systems.

The type of media used on a site depends, among other things, on the amount of data to be backed up and the time available for the backups to run.

The high capacity of some storage devices makes it practical to back up multiple file systems to one volume, but means that you must carefully consider the recovery order. Disks that are essential to the operation of the system should be backed up first, so that they can be restored as quickly as possible.

Quantity of data

The quantity of data to be backed up varies according to the nature of the data. Application images are *static*: they do not often change. But database files or individual user files are *dynamic*: they may change from day to day. Dynamic data needs to be backed up more often than static data, and you must differentiate between these data types when devising your backup strategy.

Cost and quantity of storage media

The cost and quantity of the storage media at your disposal influences the amount of backup data you keep.

The nature and capacity of the storage media may also impact on the backup window. For example, if backups are taken during lights out or operatorless periods, the backup window may be limited by the capacity and length of the backup tape.

Types of backup

The following types of backup are commonly used:

- Full
- Incremental
- Selective
- Physical
- Database

A backup strategy will use a combination of these types of backup in order to create the most comprehensive backup available, given the limitations already discussed. The backup types are defined below.

Full backups

Full backups are used to back up whole disks or file systems. A full backup is a complete copy of all the data at the time the backup was taken.

Incremental backups

Incremental backups are used to backup dynamic data between full backups. They are partial disk backups which contain only those files that have been modified since a specified date or since a previous backup.

On some platforms, you can back up incrementally from the last full backup (*cumulative*) or from the last incremental backup (*non-cumulative*).

Taking an incremental backup of all changes since the last full backup means that more data is backed up, with the associated penalties of time and media usage, but has the benefit that you can restore fully using only one full and one incremental backup.

Selective backups

A selective backup is a backup of particular files, often those contained under a particular directory structure. Selective backups are sometimes used instead of incremental backups for dynamic areas of disks.

Physical backups

A physical backup, or raw partition dump, is a backup that saves an entire volume or partition as a series of physical blocks, regardless of any file structure on the volume.

Database backups

Some database applications include their own backup system to ensure that the database is backed up in a way that enables its full or partial recreation at a later date. Database backups may be full or incremental, and if incremental, may take the form of after image journal backups.

The after image journal is a log of the database transactions undertaken: the commands issued and changes made to data. So instead of incrementally backing up the database data, you can back up the journal. Restoring the database then consists of restoring the last full backup and reprocessing all the transactions recorded since then. So when restoring from this type of backup, the data is reconstructed rather than reinstated.

A database incremental backup backs up "pages" that have changed since the last backup.

Frequency of backups

Backups should be made as frequently as is practically possible. In the ideal situation, a full backup of a disk would be made each time a file on the disk was modified. In reality, backups are made on a periodic (typically daily) basis after users have signed off the system.

In all cases, backups should be made when there is little or no risk of conflicting file access by other applications. The length of time set aside for exclusive access of data by the backup operation determines the type and frequency of the backups made.

The need to meet time criteria results in most sites using incremental and selective backups interleaved with full backups. Since extra time and media are required for full backups, they are usually done less frequently than incrementals. A typical site would make incremental backups on a daily basis and full backups on a weekly basis (usually at weekends, when there is a longer backup window available). However, the more frequently you make full backups, the more quickly and easily you can reconstruct a disk afterwards.

An alternative strategy is to balance the backup load by combining full and incremental backups for different disks on a daily basis.

You should also consider the options of taking incremental backups that record all changes since the last full backup or since the last incremental. The former option takes progressively longer to produce, but restoring takes less time.

Backup retention

In practice, it is always possible that a backup may prove unreadable or the tape may be damaged or mislaid. For this reason, it is normal practice to keep a number of *generations* of the backup. These are typically a month's worth of full backups and the last week or two of incremental backups. Tapes used for backups may be reused after the backup retention period has passed.

In addition, many sites retain one full backup from each month for a period of six or twelve months. The reasons for this are twofold:

- It enables problems such as file corruption to be traced back for more than one month
- It provides a monthly audit of the system for the year

Some sites also make one-off *check-point* backups for legal purposes. These are normally kept indefinitely and form part of an archive for audit purposes. Whilst this is an acceptable approach for checking priority systems and applications, it does have limitations. For example, tapes containing backup savesets cannot be easily refreshed to extend shelf life, and changing configurations may make it difficult to recreate the original environment from backups.

Offsite storage

The purpose of offsite storage is to keep backups in a separate location from live data and thus remove the possibility of losing both the data and the backup in the event of a major disaster such as fire or flood. The three most common practices are:

- To send tapes offsite when the next backup is taken (as above)
- To send tapes offsite after one day
- To append daily incremental backup to the tape containing the most recent full and its associated incremental backups

The last practice results in a single tape set containing the full and all subsequent incrementals up to the date of the next full backup. The tape set is moved offsite when the next full backup is made.

The choice you make depends on the criticality of the data and the cost (or even possibility) of recreating lost data when backups are also lost.

Implementing your storage management strategy

OpenMedia includes a number of features intended to make the implementation of storage management strategies simple. Among these features are:

- Pool based tape management, including multiple scratch pools
- The ability to share media between applications
- The ability to share robots between applications
- Dynamic device allocation with media type matching
- Offsite rotation
- Support for multi-volume and multi-stream savesets, which can be split into arbitrary sized segments for faster restore operations
- Preference setting for devices, for example to favor the choice of the nearest device to the tape store
- A system for generating tape movement reports and updating location statuses

OpenMedia allows you to configure a system that establishes and controls the relationships between the resources available to you.

The following chapters show how the various facilities in OpenMedia can be used to implement a typical storage management strategy.

6

Setting up and managing the object database

Initial considerations

Once you have installed and configured OpenMedia as described in Chapter 2, it is ready to use. During installation, OpenMedia creates its database and populates it with a number of database objects.

These include:

- Default locality, host, user, user accounts and privilege, to allow successful login so that you can access the database
- Standard system policies, with default values
- Standard reports
- Sample host, device, jukebox, media location and media pool objects
- Standard media type and physical type objects

No tape, saveset or segment objects are provided. These are automatically created by OpenMedia during normal running of the system, when tapes are allocated and data is written to them. See *Populating the database* in Chapter 2 for a list of the objects provided at installation.

The GUI provides you with easy-to-use object management facilities. You can use these facilities to create the objects that describe the items in your enterprise and the relationships between them, to organize your devices and tapes, and to fine-tune privileges and system policies.

Organizing the OpenMedia database

The basic building blocks used by OpenMedia are the following objects:

- Media objects:
  - Jukeboxes
  - Media locations
  - Devices
  - Tapes
  - Segments
  - Savesets

- Administration objects:
  - Localities
  - Hosts
  - Media pools
  - Physical types
  - Media types
  - System policies
  - Users
  - User accounts
  - Privileges

Media objects represent items that directly handle data, such as tapes and devices. Administration objects represent items such as hosts, users and media types. You can view and manipulate these objects via options on the **Database** and **Interrogations** menus.

Organizing the media objects in the database helps you to manage your media operations. This involves modeling jukeboxes, media locations, devices and tapes, and defining tape usage patterns in media pools. Using media pools to organize tapes is discussed later in this chapter. The other topics are discussed in Chapter 7.

Managing stored data involves organizing and tracking tapes, segments and savesets. The facilities of the **Operations** and **Interrogations** menu help you to organize and track tapes. In addition, the **Status** menu allows you to see who is connected to the server, which tapes and devices are in use, and the status of any tapes, devices and savesets currently in use. These topics are described in Chapter 8.

Administration objects are described in this chapter.

General administration

The manual *HYPERtape OpenMedia Concepts and Facilities* describes the OpenMedia database and the relationships between objects: refer to it for illustrations and examples of relationships discussed in this chapter.

The main relationships affecting general administration are:

- Media pools, media types, physical types, hosts and system policies all relate to a locality.
- A media type is always associated with a physical type.
- A tape always belongs to a media pool and has a physical type. If it is not BLANK, it will also have a current media type.

When you set up objects using the GUI, you enter these relationships using the drag and drop technique discussed in Chapter 3. For example, to enter the relationship between a host and its locality, you insert the name of the locality from the Browse "Localities" window into the name of locality field in the Host window.

Using media pools to organize tapes

OpenMedia allows you to group tapes into related media pools, with common movement cycles and retention periods. For example, you might establish media pools for different types of backup (daily, weekly and monthly) and applications (payroll system, invoicing and so on).

You can designate media pools as follows:

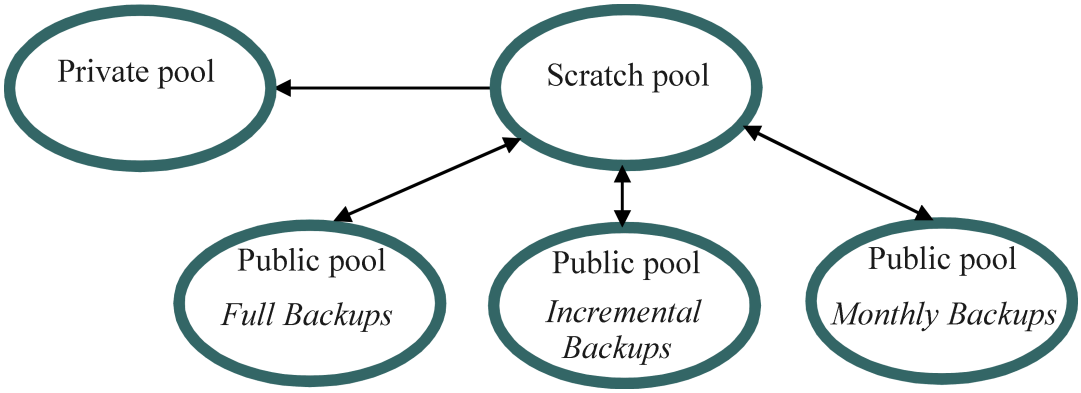
- PRIVATE
- PUBLIC
- SCRATCH

A private media pool is the simplest type of media pool. It is a set of tapes reserved for a particular usage pattern and is normally used when a special set of tapes is bought for a particular job. Only tapes from the associated scratch pool move automatically into a private pool. Tapes do not move automatically out of a private pool: you move them manually. The tapes in a private pool cycle through the usage states of blank, scratch, normal and full.

Scratch media pools contain scratch tapes, which are all available for use; these tapes are moved to other media pools when they are needed. With the aid of the OpenMedia facilities for database interrogation and scratching tapes, you can use scratch media pools to maintain an adequate inventory of reusable tapes. In most cases, this is the most economical way of using tapes because you do not need to allocate spare tapes within other media pools.

A public media pool draws tapes from an associated scratch pool when it needs new scratch tapes, and returns tapes to the associated scratch pool when all the data on them has been deleted or expired and they are scratched. This means the scratch tapes are a common resource for more than one job, so requires far fewer tapes.

The most common way to set up OpenMedia is with a single scratch pool and a series of public pools. The single scratch pool is nominated as the associated scratch pool for all the public pools.



Although you have default retentions for all tapes in a pool, you can assign and change individual saveset and tape retentions as necessary. OpenMedia maintains information about all tapes and savesets.

Manipulating locality objects

A locality object represents the area of operation of an OpenMedia service. It might represent a company branch office, or a single data center. All OpenMedia objects are tied, directly or indirectly, to a locality.

Localities are designed to be arranged in a hierarchy reflecting that of the company. The lowest level of the hierarchy is known as the neighborhood locality. User privileges relate to a neighborhood, servers serve a neighborhood and clients connect to a neighborhood. The local configuration file on each host includes the name of the local neighborhood. Changing the neighborhood name therefore has implications for the ability of clients to find the local server.

Other levels of the hierarchy include regional localities, with a single domain locality at the top. Objects such as users, physical types and media pools relate to a domain locality.

Currently, only one locality is supported, which is both a domain and neighborhood locality. The installation procedure currently creates a locality object in the database. This object cannot be deleted; neither is it necessary to add further localities. When you see the words neighborhood and domain in the GUI, they refer to this locality.

When you select the **Database/Localities** option, a selection window is displayed, which allows you to specify a wildcarded locality name.

If you click on **Add empty**, the Locality window is displayed.

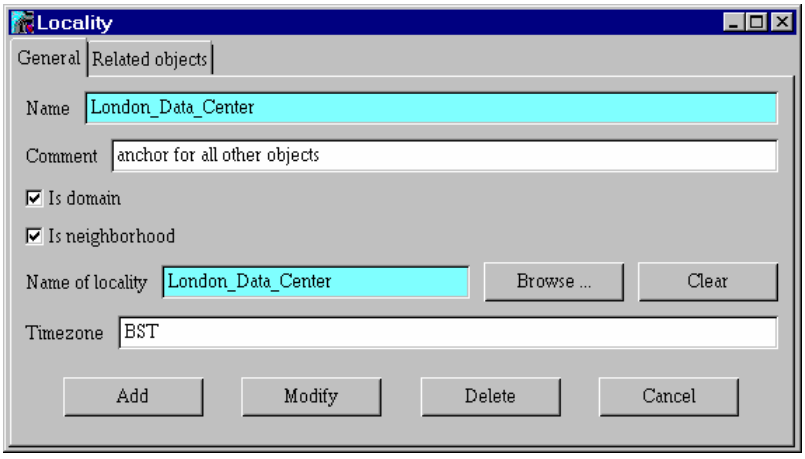
If you click on **Lookup**, the Browse "Localities" window displays a list of localities that match your entry in the name field.

Selecting a locality from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the locality in the Locality window.

Locality window

The Locality window shows details of a selected locality, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify locality objects, providing you have the correct privileges.

The Locality window has two tabs. The General tab shows general information about the locality. The Related objects tab shows the hosts that broker for the locality. From the Related objects tab, you can view individual related objects, delete existing items and add new ones to the list.



The ***Name*** field contains the name of the locality. This field is mandatory.

In the ***Comment*** field, you can record any special notes that you want to make about the locality.

The ***Is domain*** checkbox indicates if the locality is the domain locality. At the current release, this option **must** be set.

The ***Is neighborhood*** checkbox indicates if the locality is a neighborhood locality. At the current release, this option **must** be set.

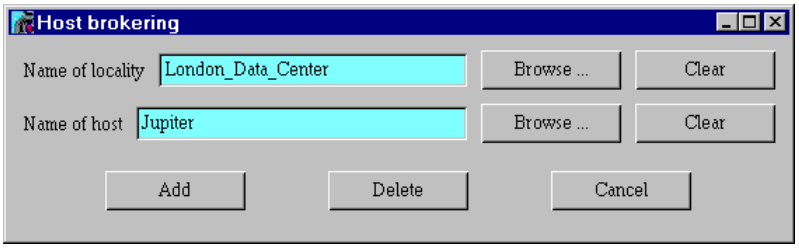
The ***Name of locality*** field contains the name of the parent locality in the hierarchy. This field is mandatory. Use the **Browse...** button to change it.

Note	*At the current release, there is only one locality for each service, so this must be the same as the name field.*

The ***Timezone*** field contains the timezone of the locality.

Viewing brokers for the locality

To view the hosts that broker the locality, click on the Related objects/Brokered by host tab in the Locality window. The tab shows a list of all the hosts that broker the locality. If you expand one of these entries, the Host brokering window appears. Using this window, you can add and delete host brokering references.



The ***Name of locality*** field shows the name of the current locality. Use the **Browse...** button to change the entry.

The ***Name of host*** field shows the name of a host that brokers the locality. Use the **Browse...** button to change the entry.

Manipulating host objects

A host object is required for any host that has tape devices or that controls devices or jukeboxes. It is not essential to create a host object for hosts that will only run the GUI or CLI, but failure to do so prevents the automatic update of the host's local configuration file and causes error messages if the OpenMedia service is started on the host.

When you select the **Database/Hosts** option, a selection window is displayed, which allows you to specify a wildcarded host name.

If you click on **Add empty**, the Host window is displayed. If you click on **Lookup**, the Browse "Hosts" window displays a list of hosts that match your entry in the name field.

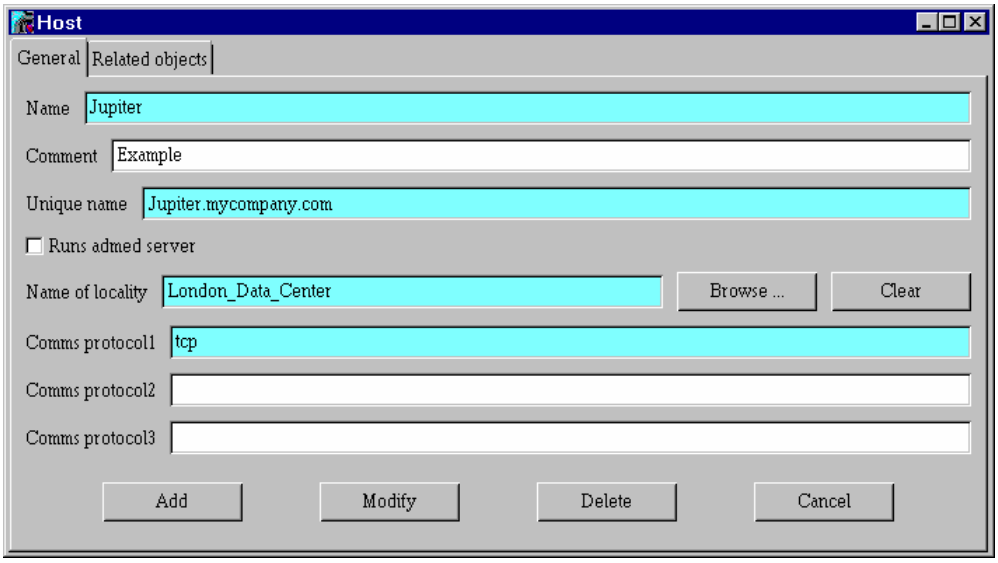
Selecting a host from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the host in the Host window.

Host window

The Host window shows details of a selected host, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify host objects, providing you have the correct privileges.

The Host window has two tabs. The General tab shows general information about the host. The Related objects tab has further tabs showing any devices accessible by the host and the localities for which the host is a broker.

From the Related objects tab, you can view individual related objects, delete existing items and add new ones to the list.



The ***Name*** field contains the name you wish to use to refer to the host within OpenMedia. Normally, this is the same as the unique name. This field is mandatory.

In the ***Comment*** field, you can record any special notes that you want to make about the host.

The ***Unique name*** is the name that your network uses for the host, which is unique within the network. Usually, this is the same as the host name used by OpenMedia. This field is mandatory.

If the host runs the OpenMedia server software, the ***Runs admed server*** checkbox should be set.

The ***Name of locality*** field contains the name of the locality to which the host is related. This field is mandatory. You cannot add a host object to the database unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

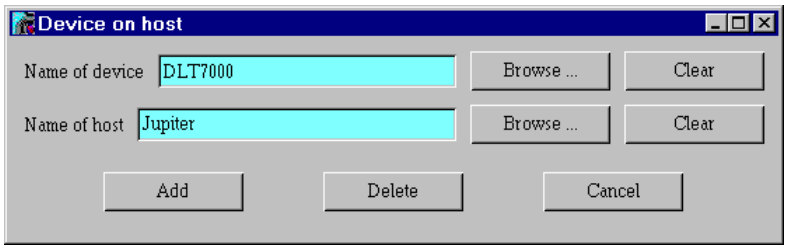
The ***Comms protocol1*** field contains the name of the communications protocol used within the service. This field is mandatory and the entry must be *tcp*.

Comms protocol2 is reserved for future use and should be left blank.

Comms protocol3 is reserved for future use and must be left blank.

Viewing devices on the host

To view the devices that the host can access, click on the Related objects/Accessible by device tab in the Host window. The tab shows a list of all the accessible devices. If you expand one of these entries, the Device on host window appears. From this window, you can add and delete device on host references.

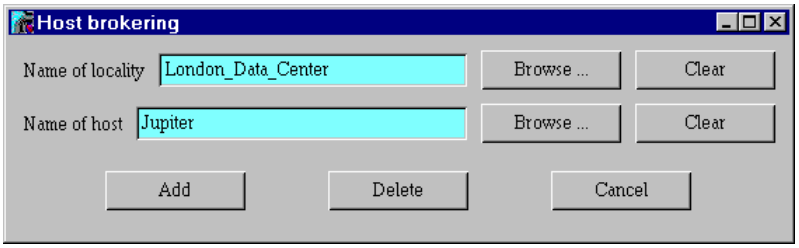


The ***Name of device*** field contains the name of a device that is accessible by the host. Use the **Browse...** button to change the entry.

The ***Name of host*** field contains the name of the current host. Use the **Browse...** button to change the entry.

Viewing localities brokered by the host

To view the localities that are brokered by the host, click on the Related objects/Brokers for locality tab in the Host window. The tab shows a list of all the localities that are brokered by the host. If you expand one of these entries, the Host brokering window appears. From this window, you can add and delete host brokering references.



The ***Name of locality*** field shows the name of a locality that is brokered by the host. Use the **Browse...** button to change the entry.

The ***Name of host field*** shows the name of the current host. Use the **Browse...** button to change the entry.

Manipulating physical type objects

The physical type represents a particular type of tape, which may then be written in several different media types. For example, a DLT tape IV can be written at several different recording densities, depending on the drive and options used to write the tape. The physical type governs whether the tape can be loaded into a drive and is mainly used when a tape is blank, to determine if a drive can initialize it.

When you select the **Database/Physical types** option, a selection window is displayed, which allows you to specify a wildcarded physical type.

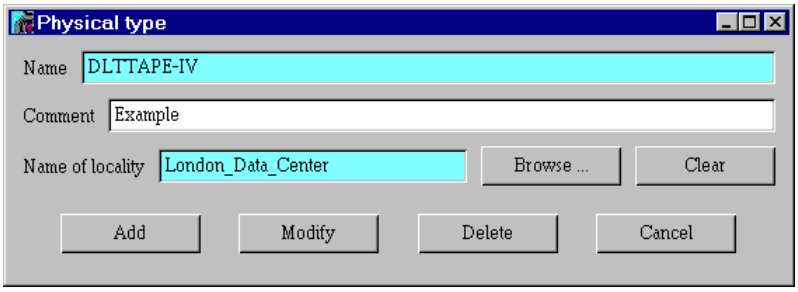
If you click on **Add empty**, the Physical type window is displayed.

If you click on **Lookup**, the Browse "Physical media types" window displays a list of physical types that match your entry in the name field.

Selecting a physical type from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the physical type in the Physical type window.

Physical type window

The Physical type window shows details of a selected physical type, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify physical type objects, providing you have the correct privileges.



The ***Name*** field contains the name of the physical type object, which must be unique within the database. This field is mandatory.

In the ***Comment*** field, you can record any special notes that you want to make about the physical type.

The ***Name of locality*** field contains the name of the domain locality to which the physical type is related. This field is mandatory. You cannot add a physical type object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

Manipulating media type objects

The media type represents how data is recorded on a tape. For example, a DLT tape IV can be recorded as DLT4000 20GB Native, DLT7000 35GB Native, DLT4000 40GB Compressed and DLT7000 70GB Compressed media types. Tapes have a media type unless they are blank, in which case they only have a physical type. Devices have a list of media types that they can read and a list of media types that they can write.

When you select the **Database/Media types** option, a selection window is displayed, which allows you to specify a wildcarded media type name.

If you click on **Add empty**, the Media type window is displayed.

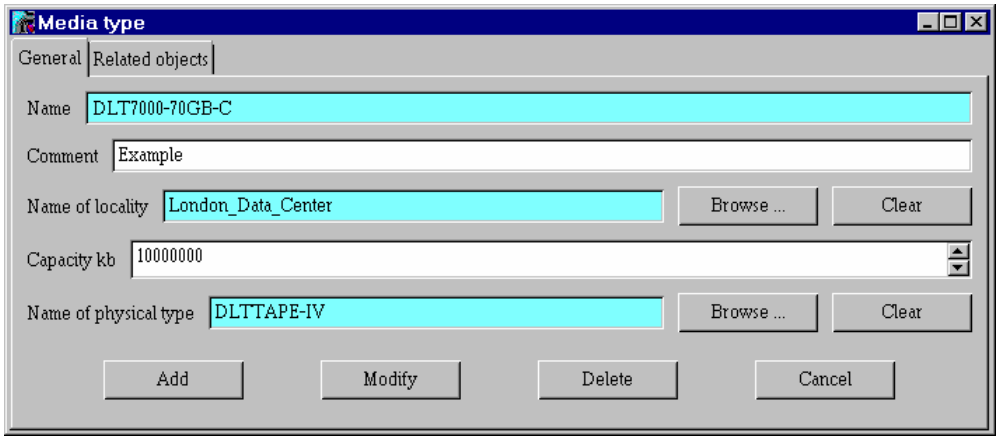
If you click on **Lookup**, the Browse "Media types" window displays a list of media types that match your entry in the name field.

Selecting a media type from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the media type in the Media type window.

Media type window

The Media type window shows details of a selected media type, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify media type objects, providing you have the correct privileges.

The media type window has two tabs. The General tab shows general information about the media type. The Related objects tab has further tabs showing the devices that can read and write the media type. From the Related objects tab, you can view individual related objects, delete existing items and add new ones to the list.



The ***Name*** field contains the name of this media type object, which must be unique within the database. This field is mandatory.

In the ***Comment*** field, you can record any special notes that you want to make about the media type.

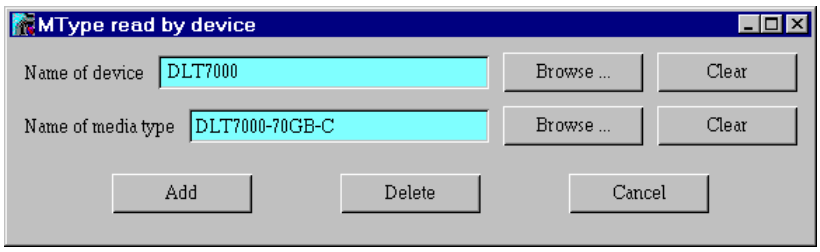
The ***Name of locality*** field contains the name of the domain locality to which the media type is related. This field is mandatory. You cannot add a media type object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

The ***Capacity*** field contains a number (in kilobytes) which tapes of this media type must have.

The ***Name of physical type*** field contains the name of the physical type object with which this media type is associated. This field is mandatory. You cannot add a media type object unless this field contains a valid physical type name. Use the **Browse...** button to change the entry.

Viewing devices that read the media type

To view the devices that read the media type, click on the Related objects/Read devices tab in the Media type window. The tab shows a list of all the devices that read the media type. If you expand one of these entries, the MType read by device window appears. From this window, you can add and delete MType read by device references.

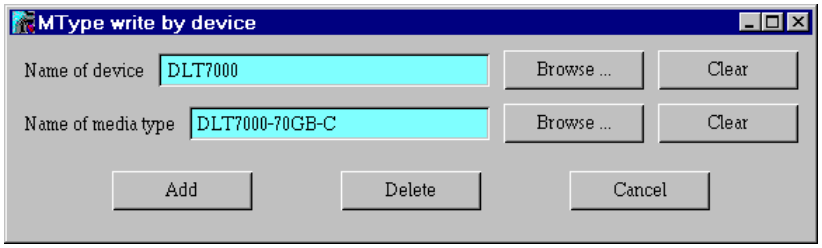


The ***Name of device*** field shows the name of a device that can read the media type. Use the **Browse...** button to change the entry.

The ***Name of media type*** field shows the name of the current media locality. Use the **Browse...** button to change the entry.

Viewing devices that write the media type

To view the devices that write the media type, click on the Related objects/Write devices tab in the Media type window. The tab shows a list of all the devices that write the media type. If you expand one of these entries, the MType write by device window appears. From this window, you can add and delete MType write by device references.



The ***Name of device*** field shows the name of a device that can writ the media type. Use the **Browse...** button to change the entry.

The ***Name of media type*** field shows the name of the current media type. Use the **Browse...** button to change the entry.

Manipulating media pool objects

A media pool defines a grouping for tapes. It allows a group of tapes to be treated identically with respect to their movement between locations and the retention of their data. Using different media pools for different types of usage allows easy automatic cycling of tapes and offsite rotations. You can add or amend defaults when you create the media pool, or at any time later.

To see the settings for any media pool, or to create a new pool, use the **Database/Media pool** option. Once you have created a pool, you can use the same option to modify or delete the pool, or to use the pool as a basis for creating further pools with similar attributes.

You can retain scratch tapes in individual pools or store them all in one or more scratch pools. A scratch pool reduces the overall requirement for tapes, because you do not need spare scratch tapes in every pool.

When you select the **Database/Media pools** option, a selection window is displayed, which allows you to specify a wildcarded media pool name.

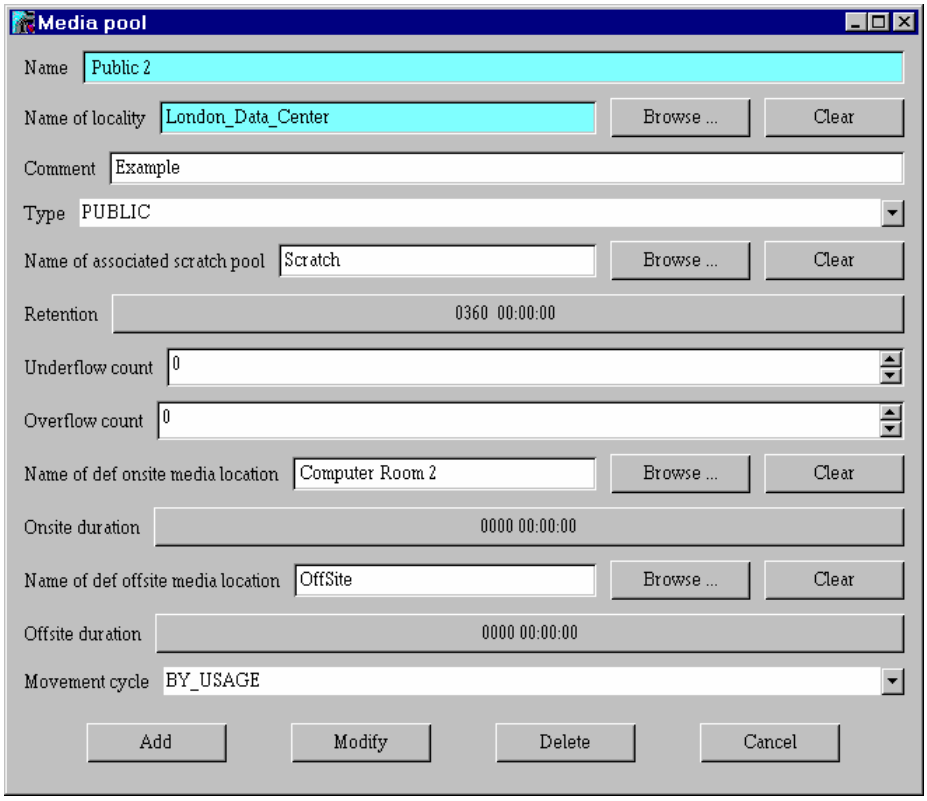
If you click on **Add empty**, the Media pool window is displayed.

If you click on **Lookup**, the Browse "Media pools" window displays a list of media pools that match your entry in the name field.

Selecting a media pool from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the media pool in the Media pool window.

Media pool window

The Media pool window shows details of a selected media pool, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify media pool objects, providing you have the correct privileges.



The ***Name*** field contains the name of the media pool object, which must be unique within the database. This field is mandatory.

The ***Name of locality*** field contains the name of the domain locality to which the media pool is related. This field is mandatory. You cannot add a media pool object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

In the ***Comment*** field, you can record any special notes that you want to make about the media pool.

The media pool ***Type*** classifies the movement of tapes to and from the media pool, as follows:

PRIVATE
: indicates that tapes can be automatically added to the media pool from its associated scratch pool, but are not automatically returned to the scratch pool when they have expired.

PUBLIC
: indicates that tapes can be automatically added to the media pool from its associated scratch pool and returned automatically to the scratch pool when they have expired. This is the most common type.

SCRATCH
: indicates that the media pool can hold only scratch tapes. Most sites have one scratch pool for all scratch tapes. You do not need different pools for different physical types of tape, although you may choose to do so to make reporting easier.

The ***Name of associated scratch pool*** field contains the name of the scratch pool from which scratch tapes are added to the media pool when they are required. Use the **Browse...** button to change the entry, or the **Clear** button to clear it. When a tape in a public media pool is scratched, it is automatically returned to the associated scratch pool. The field is generally left blank for scratch media pools.

The media pool ***Retention*** is the default period for which data is retained on tapes in the media pool. If the application writing a saveset to a tape does not set the saveset's expiration date, the media pool's retention is used to calculate an expiration date for the saveset.

The ***Underflow count*** is reserved for future use.

The ***Overflow count*** is reserved for future use.

The ***Default onsite media location*** is the name of the media location to which tapes in the media pool are assigned by default when they are onsite. Use **Browse...** to change the entry, or **Clear** to clear it.

The ***Onsite duration*** is the default period for which tapes in the media pool remain onsite when they have been written. The field is used to calculate a tape's due offsite date when the tape changes media pools, for example when a scratch tape is allocated. To ensure that tapes in the pool are never sent offsite, set this field to zero. The onsite duration of a scratch media pool is usually set to zero.

The ***Default offsite media location*** is the name of the media location to which tapes in the media pool are assigned by default when they are moved offsite. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

The ***Offsite duration*** is the default period for which tapes in the media pool are kept in storage after been sent offsite. The field is used to calculate a tape's due onsite date when the tape changes media pools, for example when a scratch tape is allocated.

The ***Movement cycle*** indicates the tape movement cycle model for the media pool. It can have the following values:

FIXED
: indicates that the movement of tapes offsite takes place after a fixed period of time in the media pool.

BY_USAGE
: indicates that movement of a tape to offsite storage takes place a given time after data is first written to the tape.

Setting system policies

System policy objects are used to set certain system defaults, such as the maximum number of times OpenMedia will attempt to load a tape and the device allocation method to be used. Default values are set for all these objects during the installation procedure.

You can add your own system policies, for example for the use of a particular client application. OpenMedia ignores any system policies that you add for your own use.

For details of the OpenMedia system policies, see *Populating the Database* in Chapter 2.

When you select the **Database/System policies** option, a selection window is displayed, which allows you to specify a wildcarded system policy name.

If you click on **Add empty**, the System policy window is displayed.

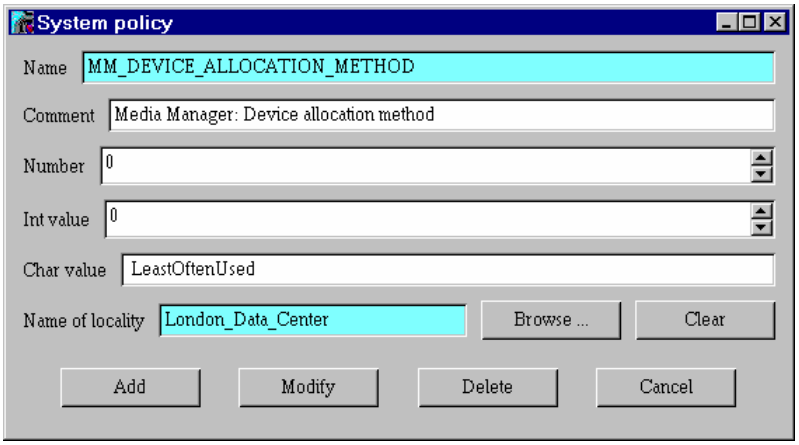
If you click on **Lookup**, the Browse "System policies" window displays a list of system policies that match your entry in the name field.

Selecting a system policy from this list (by double clicking on the object or highlighting it and clicking on **Expand**) displays details of the system policy in the System policy window.

System policy window

The System policy window shows details of a selected system policy, or provides a blank form if you are adding a new object from scratch.

From this window, you can add, delete and modify system policy objects, providing you have the correct privileges.



The ***Name*** field contains the name of the system policy object. This field is mandatory.

In the ***Comment*** field, you can record any special notes that you want about the system policy.

The ***Number*** field contains the number of the system policy. This is only used if several policies of the same name are used together to produce, for example, a table of values. This field is not currently used in any standard system policies.

Int value contains the value, if it is an integer policy.

Char value contains the value, if it is a character policy.

The ***Name of locality*** field contains the name of the locality in which the system policy applies. This field is mandatory. You cannot add a system policy object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

User administration

The main relationships that affect user administration are:

- Users, user accounts and privileges all relate to a locality.
- A user account maps a host's native user to an OpenMedia user.

The following sections explain how to manipulate user, user account and privilege objects.

Manipulating user objects

Before you can use any OpenMedia facilities, you need to log in as an OpenMedia user. When OpenMedia is first installed, a user named *manager* is created, with the password *secret*. You should login as this user in order to create your own user, user account and privilege objects.

Note	***You should change this password as soon as your installation is complete.***

You can use native or password accreditation when you log in, but your cannot log in at all unless the system can map you to a valid user object. See the section *Recovery from lost users and privileges* for information about how you can recover if you cannot login because you have lost your user objects, for example because they have been deleted.

When you select the **Database/Users** option, a selection window is displayed, which allows you to specify a wildcarded user name.

If you click on **Add empty**, the User window is displayed.

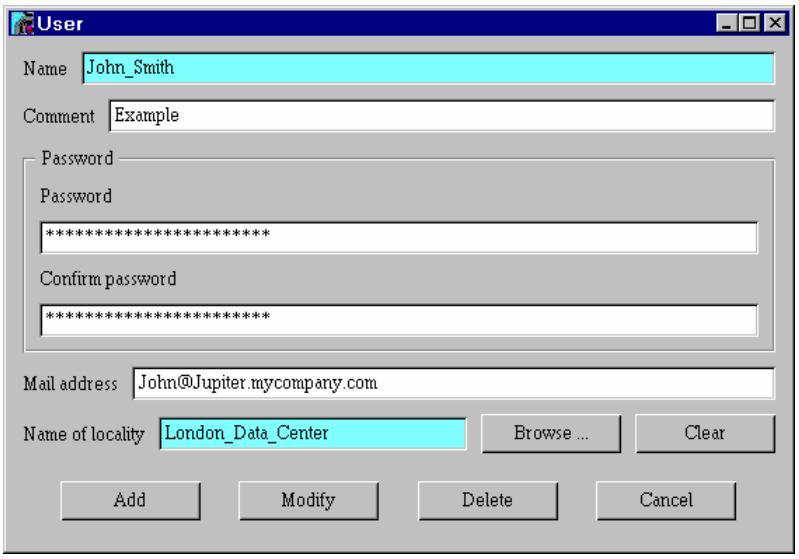
If you click on **Lookup**, the Browse "Users" window displays a list of users that match your entry in the name field.

Selecting a user from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the user in the User window.

User window

The User window shows details of a selected user, or provides a blank form if you are adding a new object from scratch.

From this window, you can add, delete and modify user objects, providing you have the correct privileges.



The ***Name*** field contains the name of the user object, which must be unique in the database. This field is mandatory.

In the ***Comment*** field, you can record any special notes that you want to make about the user.

The ***Password*** is the user's OpenMedia password, which is required when you log in using password accreditation. The password is encrypted when it is entered in the database.

The ***Mail address*** field contains the address to be used for sending email messages. This is not currently used by OpenMedia, but can be used by command files or scripts.

The ***Name of locality*** field contains the name of the domain locality with which the user is associated. This field is mandatory. You cannot add a user object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

Manipulating user account objects

User accounts allow logins from specific hosts without the need of a password. A user account object specifies a host, a native user (the username used to log in to the host) and an OpenMedia user. The installation of the server on Windows NT creates user account objects for the Administrator and SYSTEM users from the server host.

See the section *Recovery from lost users and privileges* for information about how you can recover if you cannot login because you have lost your user account objects, for example because they have been deleted.

When you select the **Database/User accounts** option, a selection window is displayed, which allows you to specify a wildcarded user name.

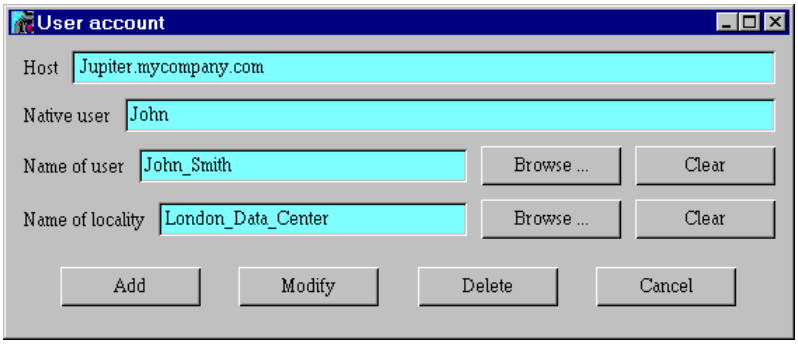
If you click on **Add empty**, the User account window is displayed.

If you click on **Lookup**, the Browse "Native host user accounts (proxies)" window displays a list of users that match your entry in the user field.

Selecting a user from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the user account in the User account window.

User account window

The User account window shows details of a selected user account, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify user account objects, providing you have the correct privileges.



The ***Host*** field contains the name of the host on which the account resides. This field is mandatory.

The ***Native user*** is the username of the user, as held on the native operating system of the host. This field is mandatory.

The ***Name of user*** field contains the OpenMedia user name. This field is mandatory. You cannot add a user account object unless this field contains a valid User name. Use the **Browse...** button to change the entry.

The ***Name of locality*** field contains the name of the domain locality with which this user account is associated. This field is mandatory. You cannot add a user account object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

Manipulating privilege objects

Privileges are required to do media management. You can assign two privileges to a user: Manager and Operator. Operator privilege allows you to answer prompts, but you require Manager privilege to manipulate tapes and device objects.

The *manager* user is automatically granted Manager privilege by the installation procedure.

See the section *Recovery from lost users and privileges* for information about how you can recover if you cannot login because you have lost your privilege objects, for example because they have been deleted.

When you select the **Database/Privileges** option, a selection window is displayed, which allows you to specify a wildcarded user name.

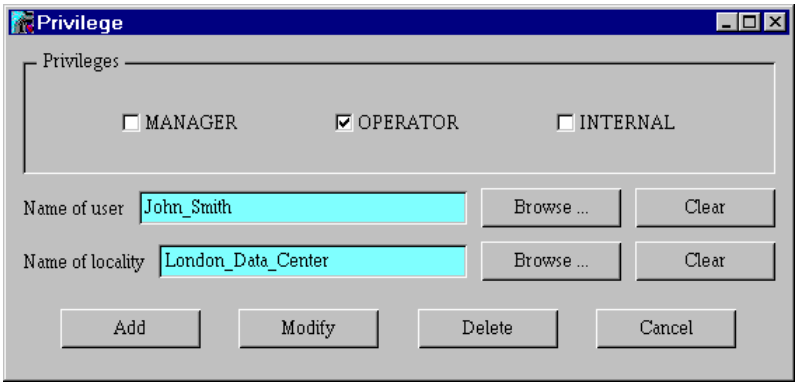
If you click on **Add empty**, the Privilege window is displayed.

If you click on **Lookup**, the Browse "Privileges" window displays a list of users that match your entry in the user field.

Selecting a user from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the privilege in the Privilege window.

Privilege window

The Privilege window shows details of a selected privilege object, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify privilege objects, providing you have the correct privileges.



The ***Privileges*** area indicates the privileges held by the user. You must have at least one privilege assigned to each user.

Manager privilege allows you access to the database and services.

Operator privilege allows you to reply to prompts.

Internal privilege is reserved for internal use.

The ***Name of user field*** contains the name of the user object to which the privilege object applies. This field is mandatory. You cannot add a privilege object unless this field contains a valid user account name. Use the **Browse...** button to change the entry.

The ***Name of locality*** field contains the name of the locality in which this privilege object applies to the user. This field is mandatory. You cannot add a privilege object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

Recovery from lost users and privileges

When you log in to OpenMedia via the GUI or CLI, you are ultimately mapped to a user object that has certain privileges.

Quoting the user name and password by logging in using password accreditation maps you directly to the user.

If you do not specify a user name and password, but log in using native accreditation, your host and native username map to an OpenMedia user account, and from there to the user.

If the database contains no privileged users, for example because they have all been deleted, use the CLI command:

```
login/database
```

to access the database and create the required objects.

If the database contains privileged users and you know a username and password, use

```
login/database username password
```

If the database contains privileged users but you cannot remember any passwords, use

`login/database/recover`

For more information about these commands, see Chapter 4.

Note	*You must be logged in to the server host as Administrator to use this facility, and you must be logged into a host that normally runs the OpenMedia server, although the server does not have to be started at the time.*

When you use the facility, you are warned and asked if you want to continue.

7

Managing media operations

Manipulating database media objects

The main relationships that affect media operations are:

- Media locations, jukeboxes and devices all relate to a locality.
- Jukeboxes are used to determine whether or not tapes in given media locations can be loaded on given devices. They can represent robots (capable of loading and unloading devices automatically) or media location and device groupings in which human operators are required to carry out manual loading and unloading operations. In robotic jukeboxes, the media location (normally just one) is the collection of slots within the jukebox.
- A media location is related to a jukebox if the tapes it contains can be loaded on a device. Offsite media locations will not be related to a jukebox if offsite tapes cannot be loaded there.
- A device is always related to a jukebox.
- A device is related to any hosts capable of accessing it.

The following sections explain how to jukebox, media location and device objects.

Manipulating jukebox objects

A jukebox is either a collection of manual tape devices controlled by the same (human) operator, or an automated tape library with devices controlled by a robot arm.

OpenMedia manages tape load and unload operations by raising prompts. The jukebox controller is a client application. It runs on hosts that control robotic jukeboxes, or that have direct access to tape drives.

The jukebox controller receives tape load, unload and eject prompts, and spawns a batch or shell script to process them. The name of the script is jukebox_driver.bat on Windows NT platforms, jukebox_driver.sh on UNIX platforms and jukebox_driver.com on OpenVMS systems. This script in turn executes a script appropriate to the jukebox model, with a name in the form name jukebox_driver_*name*, where *name* is the model name of the jukebox as held in the model field of the jukebox object.

The jukebox_driver script and a sample script called jukebox_driver_example are stored in the bin subdirectory of the OpenMedia home directory. Scripts appropriate to the jukebox models should also be stored in this directory - see the documentation supplied with your robot manager for details.

When you select the **Database/Jukeboxes** option, a selection window is displayed, which allows you to specify a wildcarded jukebox name.

If you click on **Add empty**, the Jukebox window is displayed.

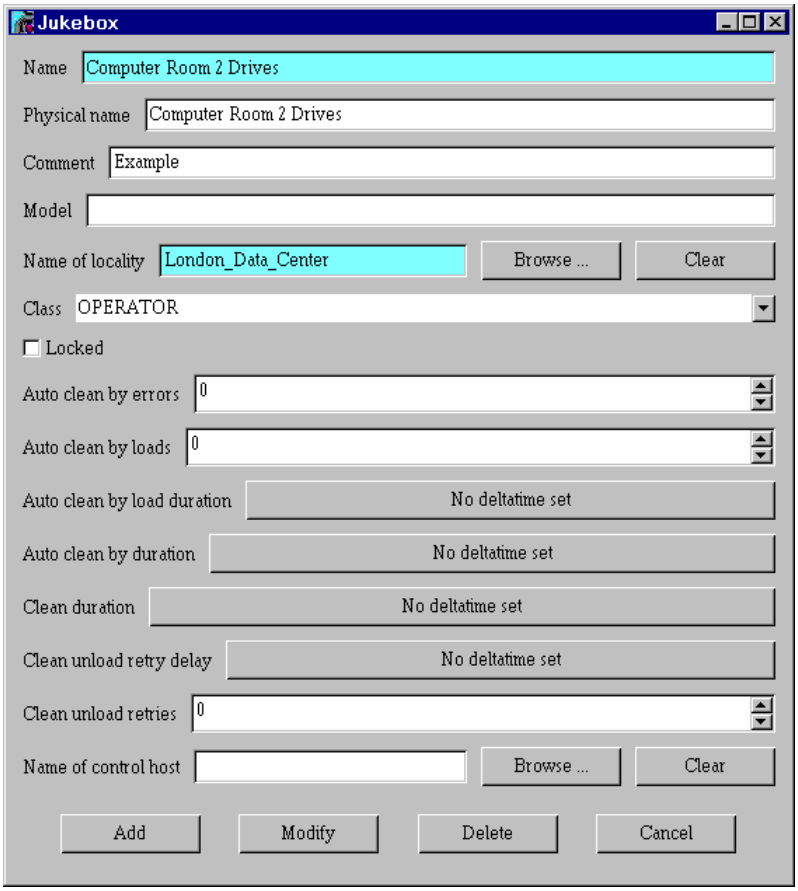
If you click on **Lookup**, the Browse "Jukeboxes" window displays a list of jukeboxes that match your entry in the name field.

Selecting a jukebox from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the jukebox in the Jukebox window.

Jukebox window

The Jukebox window shows details of a selected jukebox, or provides a blank form if you are adding a new object from scratch.

From this window, you can add, delete and modify jukebox objects, providing you have the correct privileges.



The ***Name*** field contains the name of the jukebox object, which must be unique in the database. This field is mandatory.

The ***Physical name*** is the name by which the host operating system knows the jukebox.

In the ***Comment*** field, you can record any special notes that you want to make about the jukebox.

The ***Model*** field contains the name of the robotics software that controls a GENERIC class jukebox. The field can be left blank for an OPERATOR class jukebox. The name specified here is used to identify the script that issues load. unload and eject commands to a robotic jukebox.

The ***Name of locality*** field contain the name of the neighborhood locality with which the jukebox is associated. This field is mandatory. You cannot add a jukebox object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

The ***Class*** records the nature of the jukebox. It can be one of:

GENERIC
: indicates that the jukebox is a robot.

OPERATOR
: indicates that the jukebox is a set of devices controlled by human operators.

If you set the ***Locked*** checkbox, the jukebox cannot be used.

If you enter a number in the ***Autoclean by errors*** field, the server initiates cleaning of a device in the jukebox when the number of errors reported on the device reaches that number.

Note	*Not all clients report errors to the OpenMedia server. Make sure that your clients do report error counts before using this field.*

If you enter a number in the ***Autoclean by loads*** field, the server initiates cleaning of a device in the jukebox when the number of loads on the device reaches that number.

If you set a time in the ***Autoclean by load duration*** field, the server initiates cleaning of a device in the jukebox when tapes have been loaded on it for that period of time.

If you set a time in the ***Autoclean by duration*** field, the server initiates cleaning of devices in the jukebox after the specified period.

The ***Clean duration*** field contains the approximate time that cleaning takes, after which the jukebox should attempt to remove the cleaning tape from a device.

The ***Clean unload retry delay*** field contains the time for which the jukebox should wait between failing to unload a cleaning tape and trying again.

The ***Clean unload retries*** field contains the number of times the jukebox should attempt to unload a cleaning tape after the first unsuccessful attempt.

The ***Name of control host*** field contains the name of the host that controls this jukebox. This field is used by the OpenMedia Agent when the host starts, to determine whether the jukebox controller software should be loaded on the host. It can be left blank for an OPERATOR class jukebox. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

Manipulating media location objects

Media locations are used to determine whether tapes can be loaded into devices. However, a media location may also exist outside any jukebox. A tape is loaded on a device if the tape resides in a media location which itself is in the same jukebox as the device. A jukebox may contain several media locations. This is particularly true for an operator class jukebox representing the manual tape drives that an operator can load.

When you select the **Database/Media locations** option, a selection window is displayed, which allows you to specify a wildcarded media location name.

If you click on **Add empty**, the Media location window is displayed.

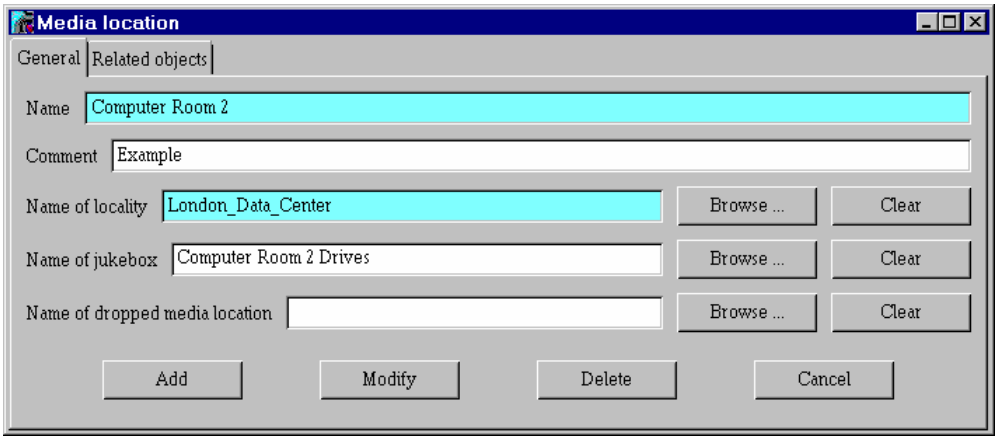
If you click on **Lookup**, the Browse "Media locations" window displays a list of media locations that match your entry in the name field.

Selecting a media location from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the media location in the Media location window.

Media location window

The Media location window shows details of a selected media location, or provides a blank form if you are adding a new object from scratch.

From this window, you can add, delete and modify media location objects, providing you have the correct privileges.



The Media location window has two tabs. The General tab shows general information about the media location. The Related objects tab shows the media location preferences for related devices.

From the Related objects tab, you can view individual related objects, delete existing items and add new ones to the list.

The ***Name*** field contains the name of the media location object, which must be unique within the database. This field is mandatory.

In the ***Comment*** field, you can record any special notes that you want to make about the media location.

The ***Name of locality*** field contains the name of the neighborhood locality with which the media location is associated. This field is mandatory. You cannot add a media location object unless this field contains the name of a valid locality. Use **Browse...** to change the entry.

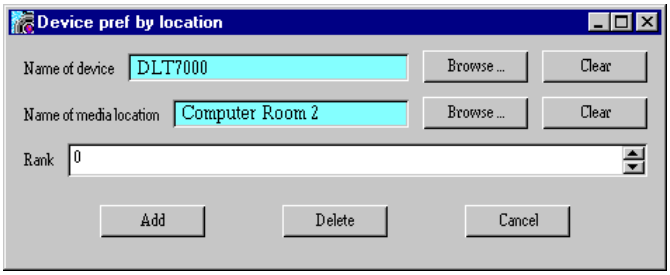
The ***Name of jukebox*** field contains the name of the jukebox with which the media location is associated. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

The ***Name of dropped media location*** field contains the name of another media location object. For example, the media location *Robot1* might identify a dropped media location of *Table_Beside_Robot1*. If a tape ceases to be reported when a jukebox inventory is performed, its media location can be altered from *Robot1* to *Table_Beside_Robot1*. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

Viewing device preferences

Device preferences allow you to prefer some devices when selecting a device in a particular media location. For example, you could tell OpenMedia to try to use a device near to a particular tape store.

To view the device preferences, click on the Related objects/Preferred by device tab in the Media location window. The tab shows a list of all the devices that are related to the media location. If you expand one of these entries, the Device pref by location window appears. From this window, you can add and delete device pref by location references.



The ***Name of device*** field contains the name of a device that is related to the media location. Use the **Browse...** button to change the entry.

The ***Name of media location*** field contains the name of the current media location. Use the **Browse...** button to change the entry.

The ***Rank*** field shows the preference of the device in the media location.

A rank of 1 denotes the most preferred device, a rank of 2 denotes a device that is preferred when no devices of rank 1 are available and so on. A rank of 0 indicates no preference.

You can have multiple devices with the same rank.

Manipulating device objects

Devices model tape drives, giving details of the capabilities of the device (read and write media types), the connectivity (hosts that can access), the control (where load and unload robot instructions can be issued). Usage statistics are maintained for each device.

When you select the **Database/Devices** option, a selection window is displayed, which allows you to specify a wildcarded device name.

If you click on **Add empty**, the Device window is displayed.

If you click on **Lookup**, the Browse "Devices" window displays a list of devices that match your entry in the name field.

Selecting a device from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the device in the Device window.

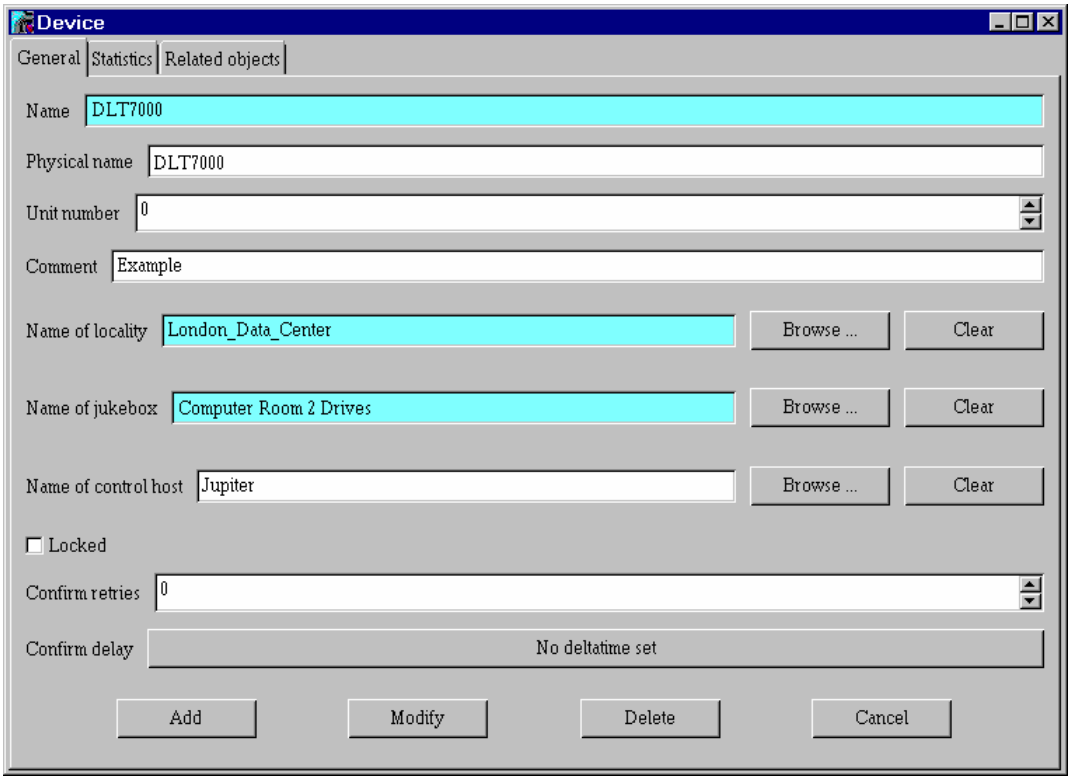
Device window

The Device window shows details of a selected device, or provides a blank form if you are adding a new object from scratch.

From this window, you can add, delete and modify device objects, providing you have the correct privileges.

The Device window has three tabs. The General tab shows general information about the device. The Statistics tab shows load and error count information for the device. The Related objects tab has further tabs showing the read and write media types related to the device, hosts by which the device is accessible and preferred media locations related to the device.

From the Related objects tab, you can view individual related objects, delete existing items and add new ones to the list.



The ***Name*** field contains the name of the device object, which must be unique within the database. This field is mandatory.

The ***Physical name*** is the name by which the device is known to the controlling host's operating system.

For UNIX systems where there are many possible names for the same device, the usual convention is to specify the no-rewind device name for highest density without compression. The client software will then either use this device name, or know how to derive alternatives.

The ***Unit number*** is the physical jukebox number of the device.

In the ***Comment*** field, you can record any special notes that you want to make about the device.

The ***Name of locality*** field contains the name of the neighborhood locality with which the device is associated. This field is mandatory. You cannot add a device object unless this field contains a valid locality name. Use the **Browse...** button to change the entry.

The ***Name of jukebox*** field contains the name of the jukebox with which the device is associated. This field is mandatory. You cannot add a device object unless this field contains a valid jukebox name. Use the **Browse...** button to change the entry.

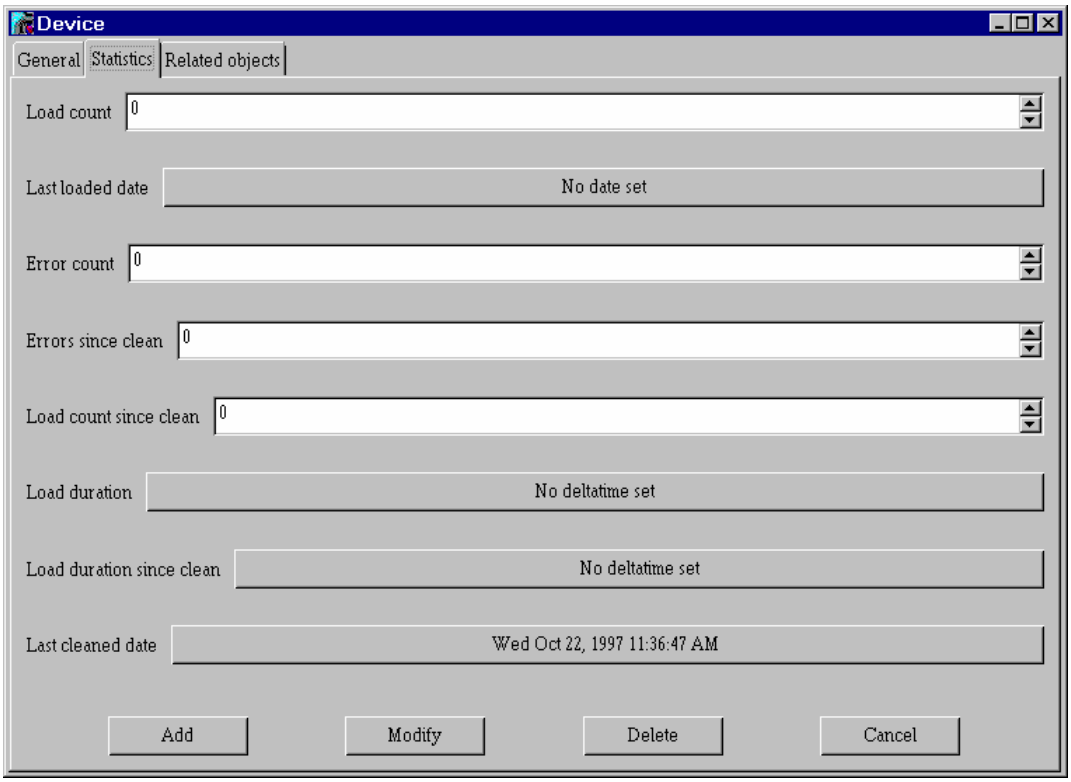
The ***Name of control host*** is the name of the host that controls the device. Prompts concerning the device are dealt with by the jukebox controller running on this host. This field is normally left blank unless drives in the jukebox are served by robots on different hosts. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

If you set the ***Locked*** checkbox, the device is not selected for use.

The ***Confirm retries*** field contains the number of times a confirm load query should be retried in the event of error.

The ***Confirm delays*** field contains the time delay that should occur between retries.

Device statistics



The ***Load count*** field contains the number of times that a tape has been loaded on the device.

The ***Last loaded date*** is the date that a tape was last loaded on the device.

The ***Error count*** field contains the number of errors recorded for the device.

The ***Errors since last clean*** field contains the number of errors recorded for the device since it was last cleaned.

The ***Load count since last clean*** field contains the number of times that tapes have been loaded on the device since it was last cleaned.

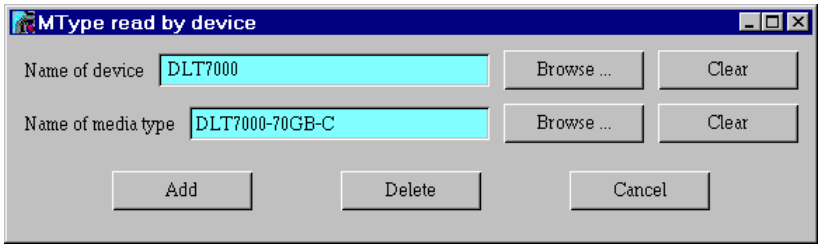
The ***Load duration*** field contains the total time that the device has had tapes loaded on it.

The ***Load duration since last clean*** field contains the time that the device has had tapes loaded on it since it was last cleaned.

The ***Last cleaned date*** is the date that the device was last cleaned.

Viewing media types read by the device

To view the media types that the device can read, click on the Related objects/Read media types tab in the Device window. The tab shows a list of all the media types that the device can read. If you expand one of these entries, the MType read by device window appears. From this window, you can add and delete MType read by device references.

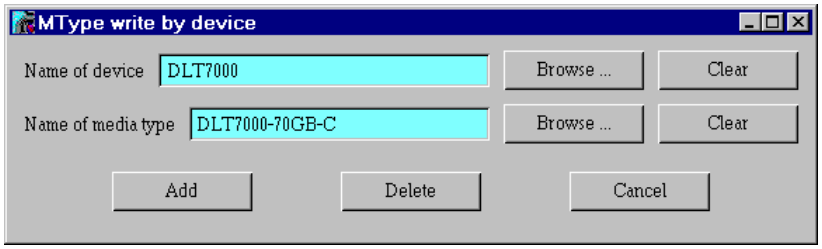


The ***Name of device*** field shows the name of the current device. Use the **Browse...** button to change the entry.

The ***Name of media type*** field shows the name of a media type that the device can read. Use the **Browse...** button to change the entry.

Viewing media types written by the device

To view the media types that the device can write, click on the Related objects/Write media types tab in the Device window. The tab shows a list of all the media types that the device can write. If you expand one of these entries, the MType write by device window appears. From this window, you can add and delete MType write by device references.

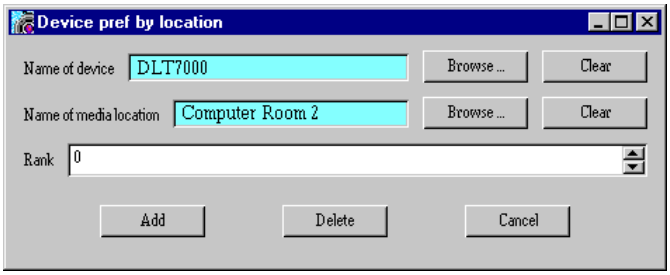


The ***Name of device*** field shows the name of the current device. Use the **Browse...** button to change the entry.

The ***Name of media type*** field shows the name of a media type that the device can write. Use the **Browse...** button to change the entry.

Viewing device preferences by location

To view device preferences in a particular media location, click on the Related objects/Preferred by location tab in the Device window. The tab shows a list of preferences for the device. If you expand one of these entries, the Device pref by location window appears. From this window, you can add delete Device pref by location references.



The ***Name of device*** field contains the name of the current device. Use the **Browse...** button to change the entry.

The ***Name of media location*** field contains the name of a media location that has a preference for the device. Use the **Browse...** button to change the entry.

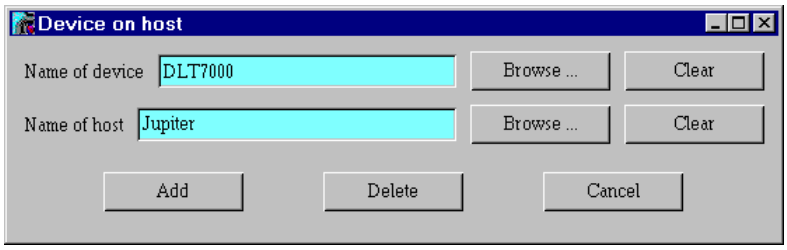
The ***Rank*** field shows the preference of the device in the media location.

A rank of 1 denotes the most preferred device, a rank of 2 denotes a device that is preferred when no devices of rank 1 are available and so on. A rank of 0 indicates no preference.

You can have multiple devices with the same rank.

Viewing hosts that access the device

To view the hosts that access the device, click on the Related objects/Accessible by host tab in the Device window. The tab shows a list of all the hosts that access the device. If you expand one of these entries, the Device on host window appears. From this window, you can add and delete device on host references.



The ***Name of device*** field contains the name of the current device. Use the **Browse...** button to change the entry.

The ***Name of host*** field contains the name of a host that can access the device. Use the **Browse...** button to change the entry.

Backing up the OpenMedia database

If you are using OpenMedia to manage all your tapes and savesets, it is essential that you make regular backups of the OpenMedia database and that you can easily locate the latest backup in the event of a disaster. Without the OpenMedia database, you will not be able to locate your backups and recover from them, so we strongly recommend that you implement a backup of the OpenMedia database as part of your general backup strategy.

You can use the batch script database_maintenance.bat in the bin subdirectory of the OpenMedia home directory as follows:

`database_maintenance backup`

backs up the database files. Note that this backup procedure simply copies the database files to the db_backup directory of the OpenMedia home directory. As this is on the same disk as the database, you should include the db_backup directory in your normal system backups.

`database_maintenance restore`

restores the database from db_backup.

`database_maintenance purge`

purges the database transaction log. This could usefully be done after the database backup.

8

Managing stored data

Tape and saveset management capabilities

OpenMedia provides a number of features that help you to manage the tapes and savesets it records in its database. These include:

- Tape scratching operations
- Saveset expiry operations
- Offsite tape handling operations
- Interrogations and reports
- Daily processing routines

OpenMedia holds objects for every tape, segment and saveset. You can access these objects directly from the **Tapes**, **Segments** and **Savesets** options on the **Database** menu. Once you have accessed an object, you can modify it if you have the correct privileges.

There is a jukebox integration script for adding tape objects to the database. For non-robotic jukeboxes, you can add tapes singly via the GUI or in bulk via the CLI ADD commands. For information about jukebox integration, see *Configuring HYPERtape OpenMedia* in Chapter 2.

In general, you should not modify the tape status or expiration date fields using the database detail windows. Instead, you should modify the fields by making higher level requests. In addition, you should not delete savesets or segments directly from the database detail windows.

You can delete tapes that are no longer required, for example because they have been sent to another site. It is usual to move these tapes to a media location reserved for this purpose, and retain the objects for a period before deleting them.

Managing the movement of tapes

There are two stages involved in managing the movement of tapes:

1 Generating reports to identify the tapes to be moved

2 Updating onsite and offsite media locations for the tapes

Identifying the tapes to be moved

OpenMedia has two interrogations to help you identify tapes that should be moved between onsite and offsite media locations:

- **Interrogations/Tapes due offsite** lists tapes due to go to the offsite store. Use this interrogation when gathering tapes together and preparing them for shipment.
- **Interrogations/Tapes due onsite** lists tapes due to be returned from the offsite store. You can print this interrogation and send it to the offsite store to help prepare tapes for shipment.

Updating the onsite media locations

Once a movement has been completed, you should update OpenMedia's onsite media locations for the tapes moved. There are two menu options that allow you to do this:

- **Operations/Mark tapes offsite**
- **Operations/Mark tapes onsite**

Onsite and offsite due dates

A tape's due onsite and due offsite dates are affected by a number of factors.

When the tape changes pools, for example when it is allocated and moves from a scratch pool to a public pool, or when it is scratched and returned to its scratch pool, the due onsite and due offsite dates are automatically reset.

When a tape moves into a non-scratch pool, its due offsite date is calculated from the time of the move plus the onsite duration of the media pool into which it moves. The due onsite date is then calculated by adding the media pool's offsite duration to the tape's due offsite date.

You can prevent tapes from ever moving offsite by setting the media pool's onsite duration to zero. Scratch pools are usually given a zero onsite duration. If you set the offsite duration to zero, the tape's due onsite date in not recalculated when the tape moves pools. This ensures that even if a tape is scratched while it is offsite, it still returns onsite on the original due onsite date. To force the return to site of offsite tapes that are scratched, you should set a short, but not zero, offsite duration for the scratch pool.

The due onsite and due offsite dates are also affected when the tape is marked onsite after returning from offsite storage. At this point, the due offsite date is cleared if the tape's movement cycle is BY_USAGE. If the media pool's onsite duration is zero (as in a scratch pool), both the due onsite and due offsite dates are cleared.

In all other cases, the due offsite date is calculated from the due onsite date plus the due onsite date plus the onsite duration of the media pool, and the due onsite date is calculated by adding the offsite duration to the new due offsite date.

Scratching tapes

Tapes should be scratched once the data they hold is no longer of value. There are two strategies for achieving this aim:

- Scratching tapes when they pass their expiration date
- Scratching tapes when their savesets are have been superseded by a given number of more recent backups

You can scratch expired tapes using the **Operations/Scratch Expired Tapes** option on the GUI, or the CLI SCRATCH command. Alternatively, you can force the expiration and removal of savesets using **Operations/Expire savesets** and **Operations/Remove savesets**, which will automatically recalculate the expiration date of the tapes on which the savesets reside. The tapes then become candidates for scratching.

Warning *You should **never** attempt to scratch a tape simply by changing its status to scratch.*

Tracking tapes

OpenMedia's tracking facilities include the ability to keep you notified of tapes that are:

- Approaching the end of their retention periods (you can interrogate by expiration date)
- Approaching the level of usage at which you would normally clean or replace them (you can interrogate by usage)
- Producing high levels of errors and requiring investigation (you can interrogate by errors)

You can also use OpenMedia's facilities for performing fast, online interrogations and printing reports to check your storage management strategy and to locate particular savesets.

Daily processing

OpenMedia provides a sample command file called daily_processing.bat in the bin subdirectory of the OpenMedia top level directory of a server host. Every day, the daily_processing script is executed at a time specified by the system policy JOB_DAILY_PROCESSING. This system policy has a character value representing the daily processing time in the form *hh*:*mm*.

When the daily_processing script is executed, it looks in the data subdirectory of the OpenMedia home directory for a command file called daily_processing_site. You should create this script if you want to define and run your own daily processing routines. OpenMedia executes this script if it finds it, passing the day of the week as a parameter (0 for Sunday, 1 for Monday, ..., 6 for Saturday). This allows you to specify certain operations for certain days of the week.

If it does not find a site defined script, the daily processing script:

- purges the database transaction logs
- scratches all expired tapes

These actions are not carried out if there is a site defined script; if you wish to perform either or both of these actions as part of your site defined daily processing routines, copy them to your daily_processing_site script.

A daily_processing log is kept in the log subdirectory of the OpenMedia top level directory.

Using media operations

The OpenMedia **Operations** menu contains a number of options that allow you to perform the following media operations:

- Scratch expired tapes
- Expire savesets
- Remove expired savesets
- Mark tapes offsite
- Mark tapes onsite
- Mark devices cleaned
- Mark devices serviced
- Display and respond to prompts

The following sections describe the operations that are available under OpenMedia. You should always use these options in preference to modifying objects directly via detail windows.

Scratching expired tapes

When all the savesets on a tape have expired, the tape can be scratched and made available for another job. You should always use OpenMedia's **Scratch expired tapes** option to scratch tapes: it has safeguards for preventing tapes from being scratched unless all the data on them has expired; and it tidies up the database, removing redundant saveset and segment objects.

You cannot scratch a tape unless its computed expiration date has passed. When this date is calculated, all the savesets on the tape are examined; the expiration date of the saveset that is due to expire last becomes the expiration date for the tape. The date is updated every time a saveset is added to the tape, and every time a saveset on the tape is expired or removed. Consequently, you cannot scratch a tape unless all its savesets have expired. If necessary, you can use the **Operations/Expire** savesets option to force the expiration of savesets.

The operation changes the status of the tape to SCRATCH and resets the computed expiration date to *No date set*. It removes from the database the tape's expired savesets and any segments relating to savesets that have expired and been removed from the tape. If the tape belongs to a public media pool, it is automatically returned to its associated scratch pool when it is scratched.

When you select the **Operations/Scratch expired tapes** option, the Expired tapes selection window is displayed. In the selection window, you can specify a date and time. Only tapes that will have expired on or before this date and time will be selected.

Click on **Lookup** to display a browser window listing the non-scratch tapes that will have expired on or before the date and time specified. From the browser window, you can examine individual tape objects by highlighting the object and clicking on **Expand** to display the tape detail window, or you can scratch one or more tapes by highlighting them in the browser and clicking on **Action**. To scratch all the expired tapes listed in the browser, click on **Select all** then click on **Action**.

You can also perform scratch operations using the CLI SCRATCH command.

Expiring savesets

The OpenMedia media **Expire savesets** option allows you to force the expiration of savesets when they are no longer required, usually because they have been superseded by more up-to-date data. When you expire a saveset, its expiration date is set to the current date and time, so making it a candidate for removal. You do not need to know the identities of the tapes on which the savesets reside, but OpenMedia recalculates the expiration dates of any tapes affected.

When you select the **Operations/Expire savesets** option, the Expire savesets selection window is displayed. In this selection window, you can specify a date and time. Only savesets that will have expired on or before this date and time will be selected. You can also restrict the selection to a set of related savesets, such as the generations of your weekly system backup, by specifying a wildcarded saveset name in the selection window.

Click on **Lookup** to display a browser window listing the savesets that match the selection criteria entered. From the browser, you can examine an individual saveset by highlighting it clicking on **Expand** to display the saveset detail window, or you can expire one or more savesets by highlighting them in the browser and clicking on **Action**. To expire all the savesets listed in the browser window, click on **Select all** then click on **Action**.

You can also expire savesets using the CLI EXPIRE SAVESET command.

Removing expired savesets

The OpenMedia **Remove savesets** option allows you to remove expired savesets from the database. You do not need to know the identity of the tapes on which the savesets reside, but OpenMedia recalculates the expiration date of any affected tapes. When you remove a saveset, the relevant segment objects are also removed from the database.

When you select the **Operations/Remove expired savesets** option, the Remove savesets selection window is displayed. In this selection window, you can specify a date and time. Only savesets that will have expired before the name and expiration date and time of savesets in which you are interested.

Click on **Lookup** to display a browser window listing the savesets that will have expired on or before the date and time specified. From the browser, you can examine an individual saveset by highlighting it and clicking on **Expand** to display the saveset detail window, or you can remove one or more expired savesets from the database by highlighting them in the browser and clicking on **Action**.

To remove all the expired savesets listed in the browser window, click on **Select all** then click on **Action**.

You can also remove savesets using the CLI REMOVE SAVESET command.

Marking tapes offsite

The OpenMedia **Mark tapes offsite** option allows you to mark tapes offsite when they have been shipped to offsite storage. The operation sets the tape's media location to the default offsite media location and its arrival date to the current date and time.

You may wish to generate a list of tapes due to go offsite at regular intervals by using the **Interrogations/Tapes due offsite** option (see *Tapes due offsite* in the *Interrogation options* section later in this chapter). After the operator actually ships all the tapes on the list, use this option to record the change of media location.

When you select the **Operations/Mark tapes offsite** option, the Tapes due offsite selection window is displayed. In the selection window, you can specify a date and time. Only tapes with due offsite dates before this date and time, and due onsite dates later than this date and time are selected.

In addition to the date, the selection window has fields for the onsite location and offsite location.

In the Offsite field, you must specify the exact value of the tape's default offsite media location, without wildcards. This is because the system checks that the current media location is not the same as the offsite location entered. If you do not specify the exact media location using the correct case for each character, no tapes will be selected.

You can enter wildcards in the Onsite field.

Click on **Lookup** to display a browser window listing the tapes that match your selection criteria. From the browser, you can examine an individual tape by highlighting it and clicking on **Expand** to display the tape detail window, or you can mark one or more tapes offsite by highlighting them in the browser and clicking on **Action**.

To mark all the tapes listed in the browser window as offsite, click on **Select all** then click on **Action**.

Marking tapes onsite

The OpenMedia **Mark tapes onsite** option allows you to mark tapes onsite when they are returned from offsite storage. The operation sets the tape's media location to the default onsite media location and its arrival date to the current date and time.

You may wish to generate a list of tapes due to return onsite at regular intervals by using the **Interrogations/Tapes due onsite** option (see *Tapes due onsite* in the *Interrogation options* section later in this chapter). When you have received all the apes on the list, use the option to record the change of media location.

When you select the **Operations/Mark tapes onsite** option, the Tapes due onsite selection window is displayed. In the selection window, you can specify a date and time. Only tapes with due onsite dates before this date and time are selected.

In addition to the date, the selection window has fields for the onsite location and offsite location. In the Onsite field, you must specify the exact value of the tape's default onsite media location, without wildcards. This is because the system checks that the current media location is not the same as the onsite location entered. If you do not specify the exact media location, using the correct case for each character, no tapes will be selected. You can enter wildcards in the Offsite field.

Click on **Lookup** to display a browser window listing the tapes that match your selection criteria. From the browser, you can examine an individual tape by highlighting it and clicking on **Expand** to display the tape detail window, or you can mark one or more tapes onsite by highlighting them in the browser and clicking on **Action**.

To mark all the tapes listed in the browser window as onsite, click on **Select all** then click on **Action**.

Marking devices cleaned

The OpenMedia **Mark devices cleaned** option allows you to reset the appropriate counter fields in one or more device objects when the devices have been cleaned. The operation resets the device's errors since last cleaned, load count since last cleaned and load duration since last cleaned fields to zero. It also sets the last cleaned date to the current date and time.

When you select the **Operations/Mark devices cleaned** option, the Devices selection window is displayed. In the selection window, you can specify a wildcarded device name.

Click on **Lookup** to display a browser window listing the devices that match your selection criteria. From the browser window, you can examine an individual device by highlighting it and clicking on **Expand** to display the device detail window, or you can mark one or more devices as cleaned by highlighting them in the browser and clicking on **Action**.

To mark all the devices listed in the browser window as cleaned, click on **Select all** then click on **Action**.

Marking devices serviced

The OpenMedia **Mark devices serviced** option allows you to reset the appropriate counter fields in one or more device objects when the devices have been serviced. The operation assumes that a device service includes a clean. It sets the last cleaned date to the current date and time, and resets all the other fields in the device's statistics tab to zero.

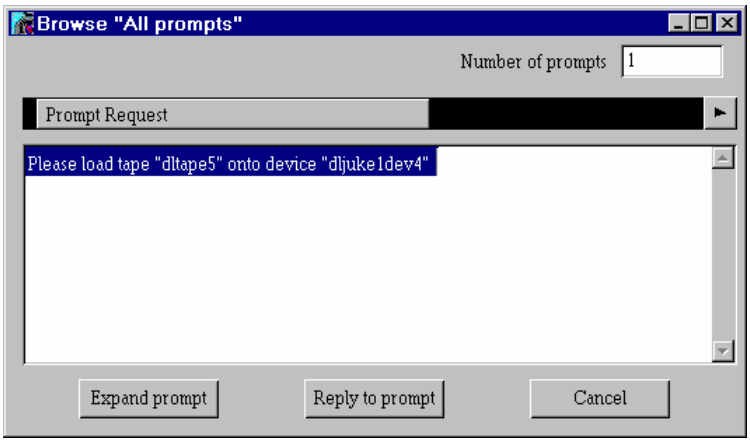
When you select the **Operations/Mark devices serviced** option, the Devices selection window is displayed. In the selection window, you can specify a wildcarded device name.

Click on **Lookup** to display a browser window listing the devices that match your selection criteria. From the browser, you can examine an individual device by highlighting it and clicking on **Expand** to display the device detail window, or you can mark one or more devices as serviced by highlighting them in the browser and clicking on **Action**.

To mark all the devices listed in the browser window as serviced, click on **Select all** then click on **Action**.

Reviewing prompts

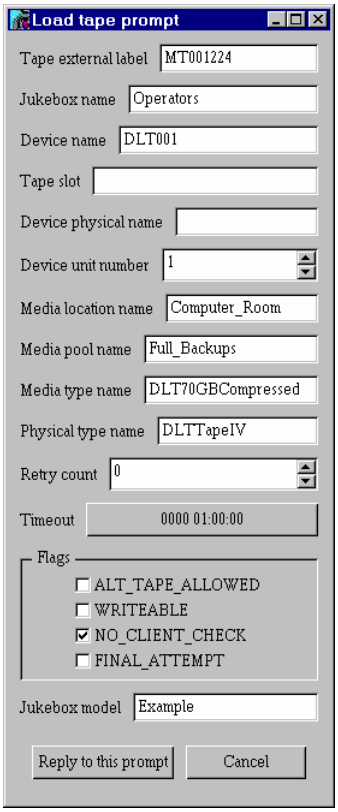
The OpenMedia **All prompts** option allows you to view outstanding prompts, and reply to them as appropriate. The option does not have a selection window. It simply displays a browser window showing all outstanding prompt. From the browser, you can expand individual prompts to view their details and issue replies.

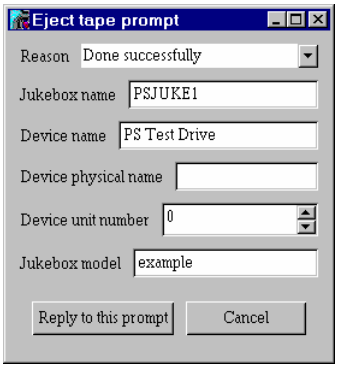


The following types of prompts may be displayed:

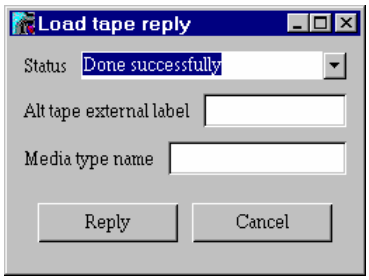
- Load
- Unload
- Eject
- Control

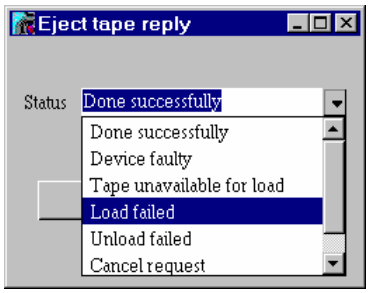
Load, unload and eject prompts concern tape operations. Each prompt relates to a particular device. Jukebox controllers reply to prompts relating to the devices they control. Other prompts may relate to devices controlled by human operators. In this case, the operator should reply to the prompt to indicate that the requested operation has been completed. Select a prompt in the browser and click on **Expand Prompt** to show information about the particular prompt.





Select a prompt in the browser and click on **Reply to Prompt**, or click on **Reply to this Prompt** in the Prompt detail window to display the prompt reply window.





Select the status you require and click on **Reply** to reply to the prompt.

Control prompts are used to shut down jukebox controllers; human operators should never reply to these prompts.

You can register for prompts and reply to prompts via the CLI as well as the GUI.

Manipulating tapes, segments and savesets

The main relationships that affect stored data management are:

- A segment represents the portion of a certain saveset on a certain tape
- A saveset can be apportioned across several tapes
- A tape can contain portions of several savesets
- When a saveset is removed, its segments remain associated with tapes until the tapes are scratched.
- A tape always belongs to a media pool.

The following sections explain how to manipulate tape, segment and saveset objects.

Manipulating tape objects

A tape object represents a tape in the enterprise. You should not normally need to modify tape objects unless there are problems with the tape and you want to mark it bad, or you want to add or change a comment. All other information is automatically updated by OpenMedia or the client application that uses the tape.

Do not change the tape status unless OpenMedia reports it as DIRTY. If you want to scratch a tape, use the appropriate options on the **Operations** menu to expire and remove the savesets if necessary, then scratch the tape.

Similarly, you should not move tapes between locations or media pools by changing the tape record via the GUI. Use the **Mark tapes offsite** and **Mark tapes onsite** options on the **Operations** menu to change the media location during the movement cycle, and use the CLI command MOVE TAPE if you want to move a tape to a different media pool.

If you replace the physical tape, you should reset the OpenMedia usage indicators such as the load and error counts.

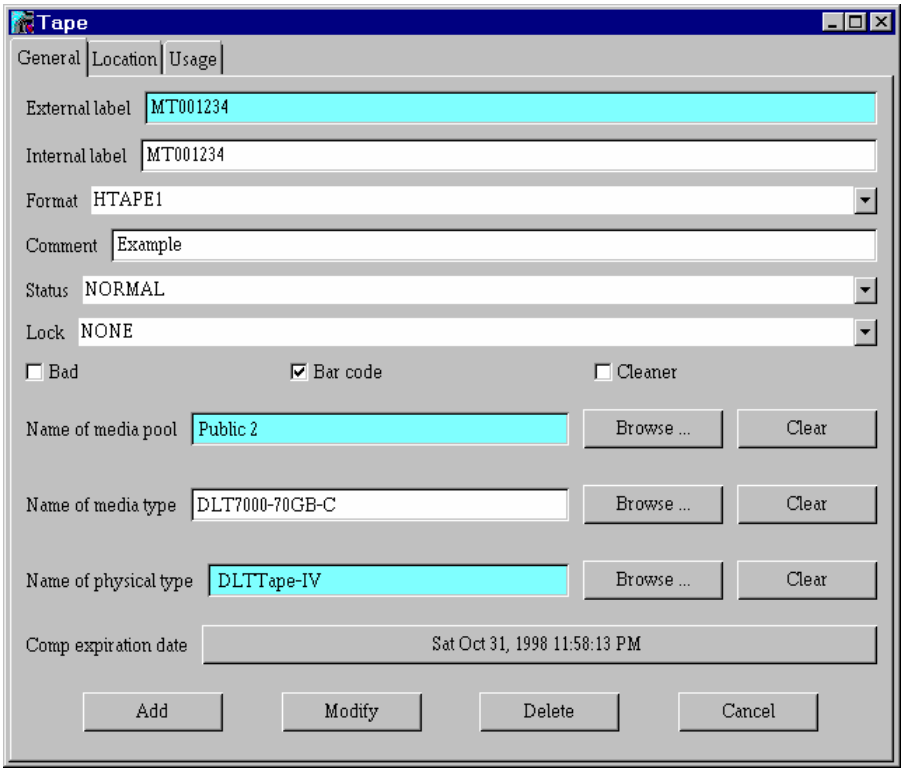
When you select the **Database/Tapes** option, a selection window is displayed, which allows you to specify a wildcarded tape label.

If you click on **Add empty**, the Tape window is displayed. If you click on **Lookup**, the Browse "Tapes" window displays a list of tapes that match your entry in the label field. Selecting a tape from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the tape in the Tape window.

Tape window

The Tape window shows details of a selected tape, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify tape objects, providing you have the correct privileges.

The Tape window has three tabs. The General tab shows general information about the tape. The Location tab shows information about the current, and default onsite and offsite media locations. The Usage tab shows usage and error count information about the tape.



The ***External label*** is the unique label designated to the tape, and written on the outside of the tape. This field is the name field of the tape. It is mandatory.

The ***Internal label*** is the label magnetically encoded on the tape.

The ***Format*** is the magnetic label type. It is one of UNLABELLED, ANSI_ASCII, ANSI_EBCDIC, IBM, NBK1, NBK2, NBK3, DB_DT, MTF, LEGATO, USER, HTAPE1 and HTAPE2 available from the pull-down list provided. HTAPE1 and HTAPE2 are required for proper integration of OpenMedia and HYPERtape Enterprise Backup.

In the ***Comment*** field, you can record any special notes that you want to make about the tape.

The ***Status*** of the tape is one of BLANK, SCRATCH, UNTERMINATED, NORMAL, FULL, DIRTY or FOREIGN, all of which are available from the pull-down list provided.

BLANK indicates a new tape that does not contain a label. SCRATCH indicates a tape that is labeled (if appropriate), but does not hold data. It may be a new or recycled tape.

When a segment is opened on a tape, OpenMedia changes the status of the tape to UNTERMINATED. The status is reset when the tape is dismounted. If a tape is left with a status of UNTERMINATED, this may indicate a problem which should be looked into before the status is modified. If the tape becomes full, its status is changed to FULL.

Tapes may also have the system assigned statuses of DIRTY and FOREIGN. DIRTY indicates that the tape contains information that is not recorded in the database, and should be investigated and possibly scratched. FOREIGN indicates that the tape format is not recognized, and the tape cannot be used.

Note	*You should not normally change this field unless it was reported as DIRTY and has been cleaned. In particular, if you change the status of an unexpired tape to SCRATCH, savesets on the tape will not be removed from the OpenMedia database. To scratch an unexpired tape, use the options on the* ***Operations*** *menu to expire and remove the savesets, then scratch the tape using* ***Operations/Scratch expired tapes****. For unterminated tapes, set the status to normal before scratching the tape.*

The tape ***Lock*** field can be one of NONE, WRITE or READWRITE. If you set the lock status to NONE, the tape can be used for reading and writing. If you set the lock status to WRITE, the tape can be read but not written. If you set the lock status to READWRITE, the tape is locked against both reading and writing.

The ***Bad*** tape indicator is meant to show tapes that have had I/O errors since the last time they were cleaned. If you set the bad tape indicator, the tape can still be used, but OpenMedia will not allow it to be reinitialized and recycled.

The ***Bar code*** indicator indicates that the tape's external label is a bar code, and should be set if the tape has a machine readable bar code.

The ***Cleaner*** indicator must be set for device cleaner tapes. Such tapes are never selected in response to tape load requests.

The ***Name of media pool*** field contains the name of the media pool in which the tape currently resides. This field is mandatory. Use the **Browse...** button to change the entry.

The ***Name of media type*** field contains the current media type of the tape. A tape cannot be used for reading or writing until it has a media type. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

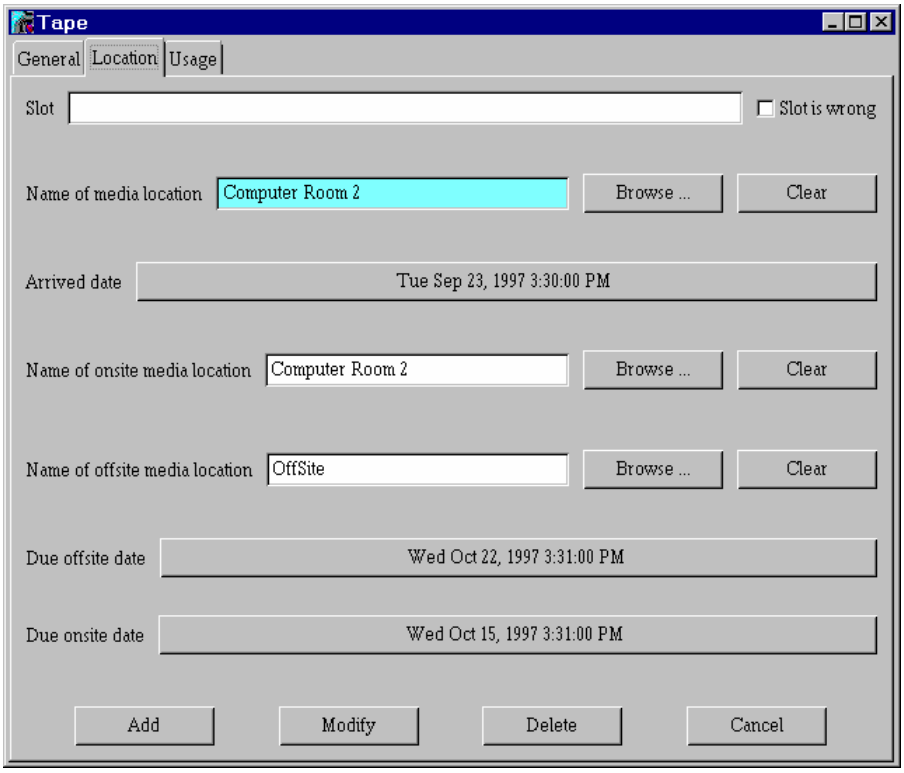
The media type may change if the tape is reinitialized.

The ***Name of physical type*** field contains the physical type of the tape. This field is mandatory. Use the **Browse...** button to change the entry.

The ***Computed expiration date*** is the date on which the tape is due to expire. It is calculated from the expiration dates of individual savesets, as they are written to the tape. If the tape is a scratch tape, the field should display *No date set*.

This field is automatically recalculated every time a saveset is added, expired or removed.

Tape location information



The ***Slot*** field is used for tapes in a robotic jukebox. It contains the slot number occupied by the tape in the jukebox.

The ***Slot is wrong*** indicator is set when the jukebox arm fails to find the tape in its allocated slot number.

The current media location is held in the ***Name of media location*** field. This field is mandatory. Use the **Browse...** button to change the entry.

The ***Arrived date*** field contains the date and time at which the tape was placed in its current media location.

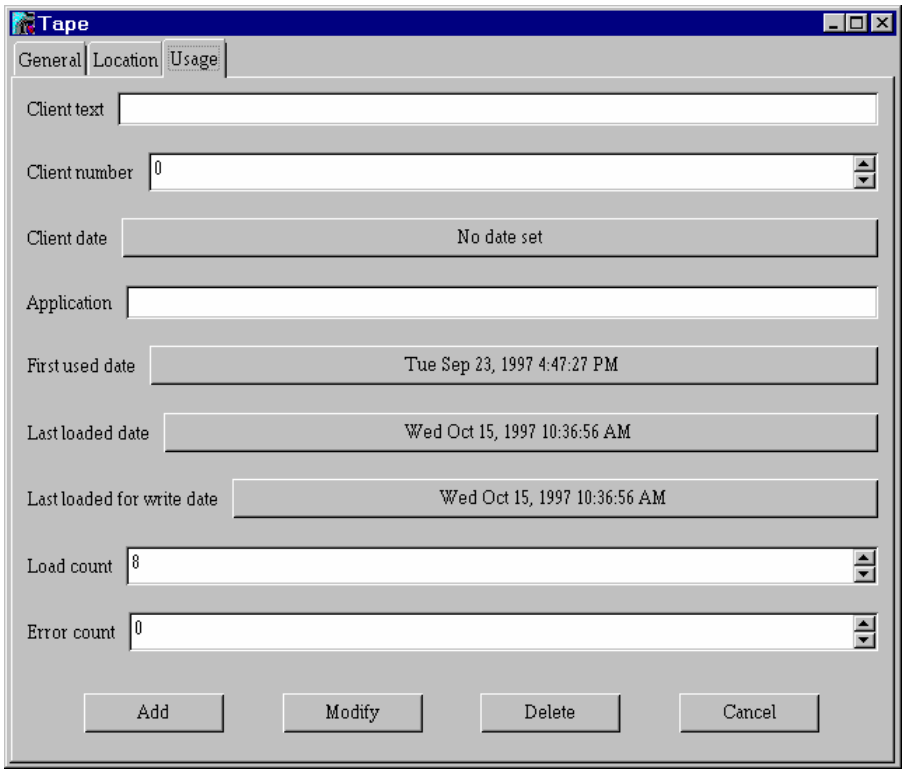
The ***Name of onsite media location*** is the media location in which the tape is stored when it is onsite. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

The ***Name of offsite media location*** is the remote facility to which the tape is sent if it is scheduled to be shipped away for storage. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

The ***Due offsite date*** is the date that the tape is due to go offsite.

The ***Due onsite date*** is the date that the tape is due to return onsite.

Tape usage information



Client text is an optional alphanumeric field, which is ignored by OpenMedia, but can be set by client applications.

Client number is an optional numeric field, which is ignored by OpenMedia, but can be set and used by client applications.

Client date is an optional date field, which is ignored by OpenMedia, but can be set by client applications and used later by clients.

The ***Application*** field is a text field that is not used by OpenMedia. It is reserved for the use of client applications, for example to contain information about the application writing the tape or the usage of the tape.

The ***First used date*** is the date the tape was first loaded after it had been initialized.

The ***Last loaded date*** field contains the date the tape was last loaded.

The ***Last loaded for write*** date contains the date the tape was last loaded for writing.

The ***Load count*** field contains the usage count, that is, the number of times the tape has been loaded.

The ***Error count*** field contains the tape error count, that is, the number of recoverable media errors encountered on this tape.

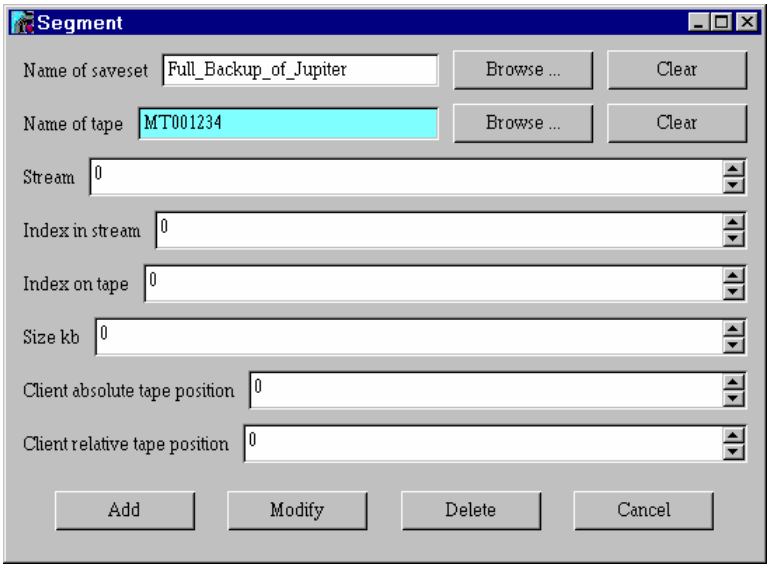
Manipulating segment objects

A segment object represents the portion of a saveset that resides on one tape, and you should not normally modify it. Updates are carried out automatically by OpenMedia and the client application. Segments are automatically removed from the database when the tape is scratched.

When you select the **Database/Segments** option, a selection window is displayed, which allows you to specify a wildcarded saveset name. If you click on **Add empty**, the Segment window is displayed. If you click on **Lookup**, the Browse "Saveset segments" window displays a list of segments that match your entry in the saveset field. Selecting a segment from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the segment in the Segment window.

Segment window

The Segment window shows details of a selected segment, or provides a blank form if you are adding a new object from scratch. Note that a segment does not have a name. From this window, you can add, delete and modify segment objects, providing you have the correct privileges.



The ***Name of saveset*** field contains the name of the saveset of which the segment is a member. Use the **Browse...** button to change the entry, or the **Clear** button to clear it.

The ***Name of tape*** field contains the name of the tape on which the segment resides. This field is mandatory. Use the **Browse...** button to change the entry.

The ***Stream number*** of the segment is supplied by the client application.

The ***Index in stream*** is the ordinal position of the segment within the stream.

The ***Index on tape*** is the segment's index on the tape. This field is automatically set when the segment is closed.

The ***Size kb*** field contains the size of the segment in kilobytes. This field is used by OpenMedia when calculating the total kilobytes currently used on the tape that contains the segment.

The ***Client absolute tape position*** is reserved for the use of the client.

The ***Client relative tape position*** is reserved for the use of the client.

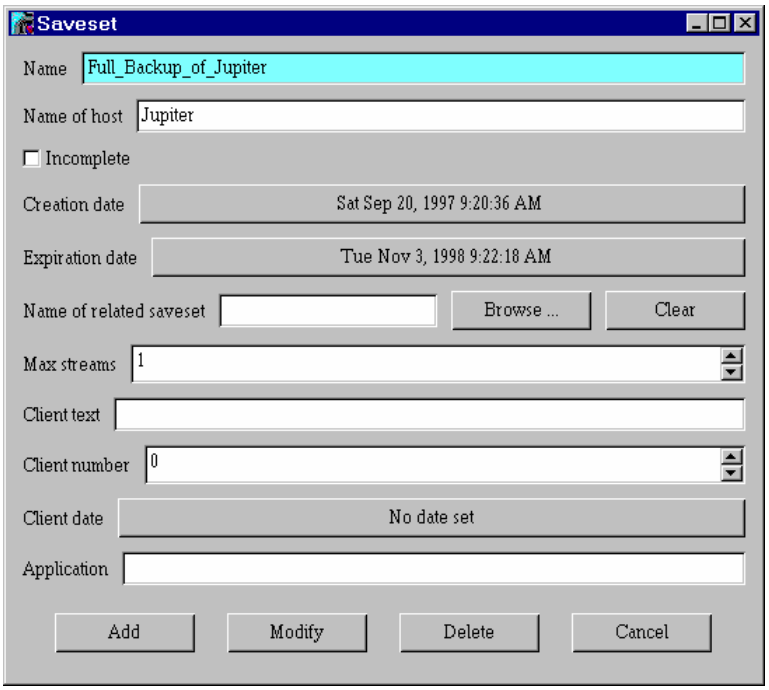
Manipulating saveset objects

A saveset object represents a saveset held on one or more tapes. You should not normally modify saveset objects. They are updated automatically by OpenMedia or client applications.

When you select the **Database/Savesets** option, a selection window is displayed, which allows you to specify a wildcarded saveset name. If you click on **Add empty**, the Saveset window is displayed. If you click on **Lookup**, the Browse "Savesets" window displays a list of savesets that match your entry in the name field.

Selecting a saveset from this list (by double clicking on the object or highlighting it and clicking on **Expand**) causes the system to display details of the saveset in the Saveset window.

Saveset window



The Saveset window shows details of a selected saveset, or provides a blank form if you are adding a new object from scratch. From this window, you can add, delete and modify saveset objects, providing you have the correct privileges.

Note	*You would add a saveset to the database only if the information was not recorded due to a tape error. You should add savesets only if you fully understand the structure of the database or are under the direction of Technical Support.*

The ***Name*** field contains the name of the saveset object, which must be unique within the database. This field is mandatory.

The ***Name of host*** field contains the name of the host to which the saveset relates. This field is used by OpenMedia when the saveset is opened.

The ***Incomplete*** indicator indicates that the system has not finished writing the saveset.

The ***Creation date*** is the date the saveset was created. The field is for information only.

The ***Expiration date*** is the date that the saveset is due to expire. If it is not set on creation, this date is calculated from the retention of the media pool of the tape containing the first segment of the saveset.

The ***Name of related saveset*** field contains the name of another saveset to which this saveset relates, for example a saveset containing an incremental backup of a file system might contain the name of the saveset containing the last full backup of the file system. Use the **Browse...** button to change the entry.

The ***Max streams*** number contains the maximum number of streams used to write the saveset. It is used when multistreaming or extraleaving.

Client text is an alphanumeric field reserved for the use of the client application that writes the saveset.

The ***Client number*** is a numeric field reserved for the use of the client application that writes the saveset.

The ***Client date*** is a date field reserved for the use of the client application that writes the saveset.

The ***Application*** field is a descriptive field intended to contain the client application that writes the saveset, or the usage of the saveset.

Interrogations

OpenMedia has a menu of interrogation options, which allow you to browse through the database and select individual objects so that you can look at them in more detail. You can manipulate the browser windows to show the information you require in the order that you want it. You can then print reports from the browser windows.

Using the Interrogations options

All the interrogations work in a similar way, as follows:

1 When you select an interrogation option from the menu, you may be presented with a selection window to refine your selection.

  The format of this window depends on the type of interrogation; for example, if you are viewing tapes by media location, the selection window prompts you for a media location. Similarly, if you are viewing devices by error, the selection window asks you to specify a minimum error count.

  You can use the wildcard characters * and ? in selection windows, as described in Chapter 3.

2 When you have entered information in the selection window, click on **Lookup** and OpenMedia presents you with a browser window containing all the items that satisfy the selection criteria you entered.

3 If you double click on an object from one of the browser windows, OpenMedia displays full details of the object you have selected. From this window, you can modify or delete the object, or add a new object, providing you have the appropriate privileges.

4 To print the contents of the browser window as a report, click on **Print** and proceed as described in the section *Printing reports* in Chapter 3.

Interrogation options

The following sections list the standard interrogations that are available under OpenMedia.

Tapes by external label

The **Tapes by external label** interrogation lists the tapes with external labels that match the entry you make in the selection window. The tapes in the browser are sorted by external label.

Tapes by internal label

The **Tapes by internal label** interrogation lists the tapes with internal labels that match the entry you make in the selection window. The tapes in the browser are sorted by internal label.

Tapes by location

The **Tapes by location** interrogation lists the tapes with media locations that match the entry you make in the selection window. The tapes in the browser are sorted by media location.

Tapes by pool

The **Tapes by pool** interrogation lists the tapes with media pools that match the entry you make in the selection window. The tapes in the browser are sorted by external label.

Tapes by usage

The **Tapes by usage** interrogation lists the tapes with a usage count greater than or equal to a number you specify in the selection window. The tapes in the browser are sorted by load count.

Tapes with errors

The **Tapes with errors** interrogation lists the tapes with an error count greater than or equal to a number you specify in the selection window. The tapes in the browser are sorted by error count.

Scratch tapes

The **Scratch tapes** interrogation lists tapes with status SCRATCH, sorted by external label. You can use this facility to determine which tapes are currently available for use. The selection window allows you to specify a physical type and media location. The tapes in the browser are sorted by external label.

Bad tapes

The **Bad tapes** interrogation lists tapes with status BAD, sorted by external label. There is no selection window.

Tapes due offsite

The **Tapes due offsite** interrogation lists tapes targeted to go offsite before a date you specify. The selection window allows you to specify an onsite media location, offsite media location, date and time. You **must** enter a valid offsite media location, without wildcards.

Tapes due onsite

The **Tapes due onsite** interrogation lists tapes targeted to be returned onsite before a date and time you specify. The selection window allows you to specify an onsite media location, offsite media location, date and time. You **must** enter a valid onsite media location with no wildcards.

Expired tapes

The **Expired tapes** interrogation lists tapes with status other than SCRATCH whose contents are due to expire to expire on or before a date and time that you specify in the selection window. The tapes in the browser are sorted by the computed expiration date. When your scratch tape inventory is low, checking the expired tape inventory should be your first step.

Tape contents

The **Tape contents** interrogation lists the segments on tapes with external labels that match the label you specify in the selection window. The segments in the browser are sorted by tape.

Device by error

The **Devices by error** interrogation lists the devices with errors greater than or equal to a number that you specify in the selection window. The devices in the browser are sorted by name.

Locked devices

The **Locked devices** interrogation lists all locked devices. There is no selection window. The devices in the browser are sorted by name.

Expired savesets

The **Expired saveset** interrogation lists savesets that will have expired on or before a date that you specify. The selection window allows you to enter a saveset name and expiration date. The savesets in the browser are sorted by name.

Monitoring media activities

OpenMedia provides a number of interrogations that help you to manage the devices and tapes currently in use. These interrogations are options on the **Status** menu. Selecting any of these options results in a list of objects in a browser window. You can customize these browsers in the usual way, and use them to produce reports - see the section *Printing reports* in Chapter 3 for more details.

Looking up reserved devices

A reserved device is a volatile object created by the server to represent an exclusive reservation of a device for manipulation by the client. A reserved tape can be loaded on a reserved device and have a saveset opened on it.

Selecting the **Status/Reserved device** option displays a browser window showing all the devices that are currently reserved for use, and the name of the user who has reserved each device. There is no selection window.

Looking up reserved tapes

A reserved tape is a volatile object created by the server to represent an exclusive reservation of a tape for manipulation by the client. A reserved tape can be loaded on a reserved device and have a saveset opened on it.

Selecting the **Status/Reserved tape** option displays a browser window showing all the tapes that are currently loaded or reserved for use and the name of the user concerned in each case. There is no selection window.

Looking up opened savesets

An opened saveset is a volatile object created by the server to represent a saveset when it is being committed to tape.

Selecting the **Status/Opened savesets** option displays a browser window showing all the savesets that are currently opened. There is no selection window.

Looking up device statuses

A device status is a volatile object created by the server to represent the current status of a reserved device or a device with a tape currently loaded on it. Device statuses are treated as extensions to devices and reserved devices while the device is in use.

Selecting the **Status/Device status** option displays a browser window showing the devices that are currently reserved or have tapes loaded on them. The browser shows the name of each device, the name of any tape loaded on it, whether the device is currently reserved, and its loaded state (UNLOADED, LOADING, LOADED or UNLOADING) . There is no selection window.

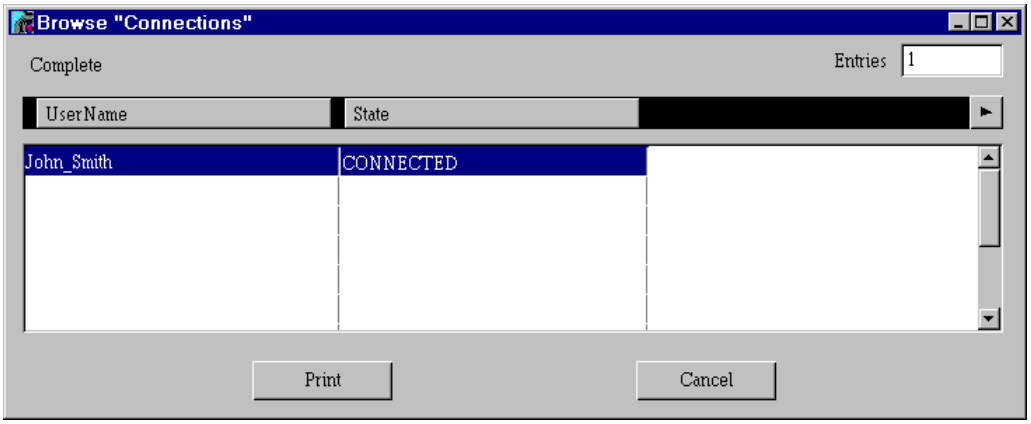
Looking up tape statuses

A tape status is a volatile object created by the server to represent the current status of a reserved tape or a tape currently loaded on a device. Tape statuses are treated as extensions to tapes and reserved tapes during these periods.

Selecting the **Status/Tape status** option displays a browser window showing the tapes that are reserved or loaded on a device. The browser shows the external label of each tape, the name of the device on which it is loaded, whether the tape is currently reserved and the number of segments it contains. There is no selection window.

Looking up connections

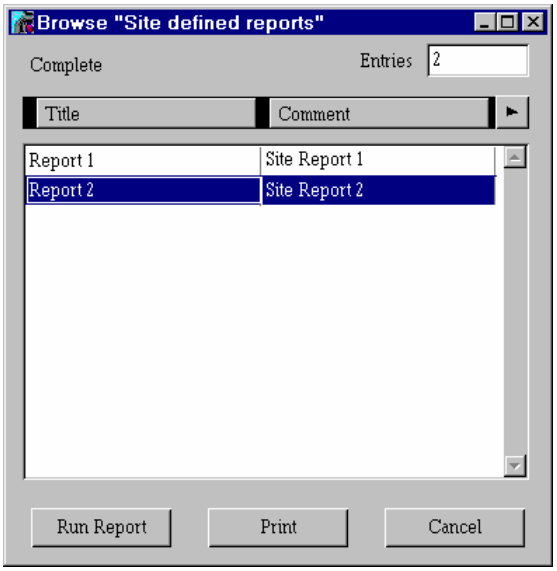
A connection is a volatile object that represents the connection between a client and a server. Selecting the **Status/Connections** option displays a browser window showing the connection status of all logged in client applications such as the GUI, CLI and jukebox controllers. There is no selection window.



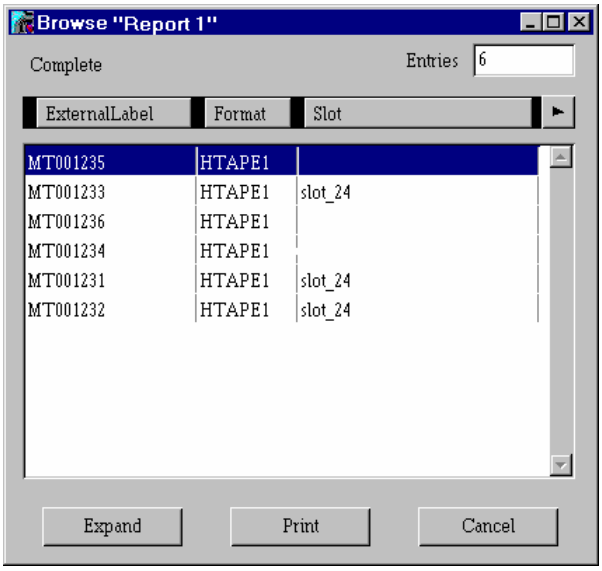
Running site defined reports

You can view and run reports from the GUI if they have been defined using the CLI and saved with the name "Site-*identifier*" where the *identifier* is any combination of characters.

Select the Site Reports option on the Reports menu to display a browser window showing all the site defined reports. Reports are shown by title and comment (although you can modify the browser in the usual way to show other fields).



To print the contents of this browser, click on **Print** and follow the procedure described in *Printing Reports* in Chapter 3. To run a report, double click on it or select it in the browser and click on **Run Report**. A selection window appears so that you can enter any selection criteria that have been tagged in the report definition. Enter your selection criteria and click on Lookup to display a browser window showing the objects that match the selection criteria.



You can add, delete and rearrange the columns, expand individual objects in the window and print the report in the normal way, see *Using browsers* in Chapter 3 for details.

A

HYPERtape OpenMedia server

Operation of the server

Most of the OpenMedia operations are undertaken by the server. Several servers may be installed in an enterprise, but they do not share any knowledge of that enterprise with each other. If multiple servers are installed, clients can ask for different services and may be connected to different servers. Each server provides the media management for a particular locality.

The initial connection between clients and server is established through a service broker: clients ask for a service rather than a particular server and are connected to any server offering that service. Clients use the API to create requests. After executing a request, the server returns a response.

Startup and shutdown

A server is normally started by the OpenMedia Agent when the host is booted, then runs continuously. When a server starts, it assesses the services it is to offer and determines which locality it is to serve. It then:

- Reads its local configuration file
- Opens its database
- Refreshes its local configuration file from the database
- Registers its service(s) with service broker(s)

To shut down a server, stop the OpenMedia service as described in *Starting and stopping the OpenMedia service* in Chapter 2. A server can be shut down by an instruction from the CLI or a client application, although this is not recommended. It can shut down in a clean manner, or perform an emergency shutdown.

When shutting down cleanly, the server rejects all new requests except for an emergency shutdown, then waits for all short-lived requests to complete. Once the requests are complete, the server:

- Rejects all new requests
- Cancels all current requests
- Destroys client contexts
- Releases reserved devices and reserved tapes.

An emergency shutdown follows the same procedure without waiting for requests to complete.

Security and privilege

Accreditation establishes the security credentials of the client user with the server, by identifying a particular OpenMedia user.

Recognizing a known user is not in itself sufficient to allow access to OpenMedia services. Known users are granted privileges in localities, and it is the user's privilege within a locality that is checked before services are provided. OpenMedia supports MANAGER privilege, which allows the user access to any service and any object in the locality, and OPERATOR privilege, which allows the user to register for and reply to prompt in the locality and to shut down servers.

To shut down a server, a user requires OPERATOR privilege in the locality for which the server offers services.

Error and disaster recovery

If a process is lost, restart OpenMedia. This will restart any missing components. As the OpenMedia Agent automatically attempts to start all components required on the host, error messages may be generated for components already running. You can ignore these messages.

If a process hangs, shut down and restart the OpenMedia service.

Lost connection to a client

The connection to a client can be lost for several reasons. For example, the communication line can drop, the client host can crash or the client software could fail. Connection can also be naturally broken and resumed during a sequence of requests. When the connection changes state, attempts at reconnection are left to the client. All the server does is to retain the client's context for a period of time determined by the system policy NET_RECON_TIMEOUT. From the server's point of view, there are only two possible outcomes from the client's attempts at a reconnection in the time allowed:

- *Success* All context is retained, including outstanding requests, change and prompt registrations, reservations and opened savesets. The client can continue as though the connection was never lost or broken.
- *Failure* The server destroys the client's context, closes any opened savesets or opened segments, releases reserved devices and reserved tapes and cancels change and prompt registrations.

Abnormal termination of the server

Abnormal terminations can occur if the server host crashes or the server software fails. To alleviate the problems that this can cause, the server holds client context in volatile objects and not in the database. For example, device reservations are volatile objects, and do not involve transient changes to device fields in the database.

A server may fail when there are loads and unloads outstanding, for example during a backup. As information about reserved tapes is held in volatile objects, there is no effect on the OpenMedia database other than usage counts being out of date. Any backup on progress will probably fail.

B

HYPERtape OpenMedia messages

Message format

The following list shows the message codes output by OpenMedia with explanations. You may see them when working with the GUI or CLI, or they may be returned by a client application.

The messages are in alphabetical order by message code, which is usually an abbreviation of the message text. The message text describes the condition that produced the message. The severity level is one of: Success, Info, warning, error and fatal.

Message list

`ADMINONLY 0BBC858A Error  Option available only to native admins`

The attempted operation is only available when running from a native administrator account.

`ALLALREADYRES 0BBC8902 Error  All such tapes and/or devices already reserved`

A reserve and load tape on device request has failed because all suitable tape and device combinations were already reserved, so could not be offered to the client. The client could wait for a time then reattempt the request.

`ALRDYRUN 0BBC869A Error  Facility is already running`

This error is returned by the startup procedures of various managers when they are already started. It is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`ALTTAPENOTPERM 0BBC80CA Error  Alternate tape not permitted`

An operator has supplied an alternate tape when the requested tape was required.

`AMBFLDNAM 0BBC833A Error  Ambiguous field name '__'`

The specified field name is ambiguous.

`AMBKEYNAM 0BBC8422 Error  Ambiguous keyword '__' for '__'`

The specified enumerated name is ambiguous.

`AMBOBJNAM 0BBC8332 Error  Ambiguous object name '__'`

The specified object name is ambiguous.

`AMBSEGINF 0BBC848A Error  Ambiguous segment information`

A segment created with the specified parameters would not be uniquely identifiable. Use a different stream.

`ASPINS 0BBC891A Error  Associated scratch pool is not SCRATCH`

A pool associated as a scratch pool to another pool has a type other than SCRATCH.

`ATCBUI 0BBC80D2 Error  Alternate tape cannot be uniquely identified`

The alternate tape supplied by an operator does not exist in the database, or the details given by the operator match more than one tape.

`BADFLDNUM 0BBC87C2 Error  Invalid field number`

An invalid field number has been specified. Legal field numbers are defined in the fit public header. API: Check your code. Otherwise, this is an internal error: note the operation in progress, save any trace information, and submit a fault report.

`BADJUKINF 0BBC84C2 Error  Bad record (__) in Jukebox Information File`

The Jukebox Information File contains an invalid record.

`BADMODCOUNT 0BBC82E2 Error  Record has invalid modification count`

A database update operation has not been carried out because the modification count in the object did not match the modification count in the database. This may be because the object was updated after you browsed it but before you updated it. If returned from any operation other than an update, it may indicate a product internal error: note the operation in progress, save any trace information, and submit a fault report.

`BADOBJTYP 0BBC87B2 Error  Invalid object type`

An invalid object type has been specified. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`BADREFINT 0BBC82F2 Error  Referential integrity violation`

Database objects are related together through references (*UuidOf* fields). You cannot modify the database in a way that leaves an invalid reference. For example, many objects are related to a Locality object through a *UuidOfLocality* field and deleting the Locality object would cause a referential integrity violation.

`BADREQTYP 0BBC812A Error  Bad request type`

The specified request is the wrong type for the attempted operation.

`BADSORTNUM 0BBC87F2 Error  Invalid sort number`

An invalid sort number has been specified. Legal sort numbers are defined in the fit public header. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`BADTAPEINFO 0BBC80EA Error  Illegal tape details`

The tape information supplied by a client has not been successfully passed to the admed server.

`BADTAPELOCK 0BBC822A Error  Bad tape lock`

A segment cannot be opened on the specified tape because the tape is locked.

`BADTAPESTS 0BBC8222 Error  Bad tape status`

A segment cannot be opened on the specified tape because the status of the tape is not SCRATCH or NORMAL.

`BATSIZSET 0BBC82FB Info  Batch size too large or not specified; has been set to maximum`

The API imposes a maximum batch size (*InterCnt*) on requests that return lists of items. The request either did not specify an *InterCnt* or specified one that was too large. The *InterCnt* in the request has been set to the maximum permitted.

`BUFLENSTOOSMALL 0BBC8732 Error  Lengths buffer too small (internal error)`

An internal lengths buffer is too small. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`BUFTOOSMALL 0BBC873A Error  Internal buffer to small`

An internal buffer is too small. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`CACDBRMIS 0BBC850A Error  Cache database record __ missing`

A required cache record has not been found in the database. This message may appear during product installation, when it can be ignored. During normal running, it indicates that important records required for the proper functioning of the software no longer exist in the database.

`CACRDERR 0BBC86E2 Error  Error reading cache`

A record cannot be read from the internal cache. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`CACRFRERR 0BBC85B2 Error  Cache refresh error`

An error has occurred during cache refresh. Check the trace log for more information.

`CANCEL 0BBC802A Error  Cancel operation`

This status should be returned from a *NoteProc* by an API client: it directs the component that called the *NoteProc* to cancel the operation.

`CANCELED 0BBC8020 Warning  Request or operation canceled`

The specified request or internal operation was canceled, probably before completion.

`CANTCANCEL 0BBC86EA Error  Target request is of a type that cannot be canceled`

The load manager has been asked to cancel a task that can't be canceled This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`CANTSCARFT 0BBC813A Error  Cannot scratch foreign tape`

Tapes with a status of FOREIGN cannot be scratched.

`CANTSCRACL 0BBC8142 Error  Cannot scratch a cleaner tape`

Cleaner tapes cannot be scratched.

`CBPCOMFAI 0BBC860A Error  CLI background process communications failure`

Communication with the background process failed. If MMCLI is invoked again, a new background process will be created. Check the trace log to determine why the CLI background process (MMCLIMAIN) failed.

`CBPCXFINV 0BBC856A Error  CLI background process context file invalid`

The context file created by the background process is not in the expected format. Check that MMCLI.EXE and MMCLIMAIN.EXE are the same version, i.e. you do not have a mixed version installation or two installations confusing each other.

`CBPNOTRSP 0BBC85CA Error  CLI background process not responding`

The CLI front end started the background process, but it has not created the expected context file. Check the current temporary (scratch) directory for file permissions, etc. Check that MMCLIMAIN.EXE is in the same directory as MMCLI.EXE.

`CHGALRREG 0BBC846A Error  Already registered for changes`

In the CLI, you cannot register for changes more than once.

`CHGNOTREG 0BBC8472 Error  Not registered for changes`

You cannot deregister for changes if you are not currently registered.

`CLICRITXTAPE 0BBC80DA Error  Client criteria excludes tape`

The tape supplied by an operator is not allowed under the criteria supplied by the client.

`CLIMISHLP 0BBC8562 Error  Unable to access help file`

The CLI help file ENGLISH_HELP.TXT could not be found. It is located via the environment variable OMM_CLI_HELP (which must specify the full pathname), or failing that in the same directory as the MMCLIMAIN executable file.

`CLINOTSTA 0BBC885A Error  Client not started`

The client startup routine has not yet been called. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`CLIRFFMIS 0BBC845A Error  Required field __ missing`

The specified field is required, but is missing in the object.

`CLIRFOMIS 0BBC8462 Error  Required field(s) missing from __`

One or more required fields are missing from the specified object. Use display *object* to identify required fields in an object (they are marked as mandatory).

`CLSINPROG 0BBC82BA Error  Anchor is already being closed`

EndConnection has already been called on this anchor. Further close calls are ignored. API: Correct your code. Otherwise, this is an internal error: note the operation in progress, save any trace information, and submit a fault report.

`CNTCONBRK 0BBC8242 Error  Cannot connect to service broker`

The program was unable to connect to an admed server because it was unable to connect to the service broker. Check the trace log for more information. Check that the LOCALBROKER entry in the local configuration file specifies the host that runs the service broker.

`CNTCONSRV 0BBC8252 Error  Cannot connect to server`

The program failed to connect to a server. Admed servers are located by neighborhood name, and the default neighborhood is taken from the local configuration file (keyword LOCALHOOD). The CLI command login -list shows the available neighborhoods.

`CNTFINVER 0BBC8692 Error  Cannot find version in service attribute list`

There is no version attribute in the service attribute list This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`CNTSTRCBP 0BBC8512 Error  Cannot start CLI background process`

The CLI front end was unable to start the background process. Check that MMCLIMAIN.EXE is in the same directory as MMCLI.EXE.

`CORTEX 0BBC8672 Error  Please note the operation currently in progress and submit a fault report`

Note the operation in progress, save any trace information, and submit a fault report.

`CRECRITFAIL 0BBC85BA Error  Failed to create criteria`

An operation requiring the creation of criteria has failed to create them. Check the trace log for more information. If the problem seems to be lack of resources, allocate more virtual memory to the program and restart.

`CRICHKERR 0BBC879A Error  Client's criteria is not legal`

The client criteria have been checked against a handler's illegal criteria and have failed the check. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`CSGTAPSTS 0BBC847A Error  Request specifies bad tape status`

A request to close an opened segment specified an illegal value for the status of the Tape. Legal statuses are NORMAL, FULL, and UNTERMINATED.

`CTXNOTEMP 0BBC8492 Error  Cannot END, you must RELEASE tapes/devices or CLOSE savesets/segments`

The CLI has rejected an END command because the context indicated has a reserved tape, reserved device, opened saveset or opened segment. Release or close these.

`CTXSAVOPN 0BBC88BA Error  The specified context already has an open saveset`

A CLI open saveset command cannot be performed because the specified context already has an opened saveset.

`DBACCFAIL 0BBC85C2 Error  Failed to connect to, open, prepare or disconnect from the database`

The database has its own server, to which the database handler connects: this may have failed to start. The database name is an encoding of the product release: this may not match. The database server is prepared for use by the database handler: this process may have failed.

`DBBADFLDNUM 0BBC87D2 Error  Invalid field for database lookup`

The field number is legal but the field type is not *database* or *mmref*, so it cannot be looked up in the database. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`DBBADOBJTYP 0BBC883A Error  Invalid object type for database operation`

A database request has been made on a non-database object. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`DBBADREQUEST 0BBC8742 Error  Invalid database request`

An unknown, unsupported request has been made to the database component. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`DBBADSORTNUM 0BBC8802 Error  Invalid sort for database lookup`

The sort number is legal but its type is not *database field* or *key*, so it cannot be used to sort database objects. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`DBBADTID 0BBC875A Error  Invalid task id`

An invalid task id was supplied to a database suspend, resume or cancel This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`DBBUFTOOSML 0BBC871A Error  Buffer too small (internal error)`

The database component has detected that an internal buffer may be too small. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`DBDEACALR 0BBC887A Error  Database already deaccessed`

You cannot deaccess the database twice.

`DBDEACCESS 0BBC8312 Error  Access to the database suspended for maintenance`

Access to the database files is suspended whilst maintenance takes place. The operation you requested could not be paused or queued during the maintenance period, so it has been aborted.

`DBDEACCESSED 0BBC888B Info  Database has successfully deaccessed`

The database has been deaccessed. Any suspended lookup requests have been canceled. Other requests accessing the database have been completed before the database was deaccessed. Any further requests that access the database will hang until a database reaccess occurs.

`DBHNDFAIL 0BBC8712 Error  Db-handler doing database operation failed`

The db-handler carrying out this database operation terminated unexpectedly. The operation can be requested again and will be handled by another db-handler. Note the operation in progress, save any trace information, and submit a fault report.

`DBMAKSQLERR 0BBC8752 Error  Error in derived SQL select statement`

Data returned from the SQL select statement generator was too long. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`DBNOTCLOSED 0BBC8722 Error  Database not closed`

An operation that is only legal on a closed database has been attempted. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`DBNOTDEAC 0BBC8882 Error  Database not deaccessed`

You cannot reaccess the database if it has not been deaccessed.

`DBNOTDEACCESSED 0BBC831A Error  Database not de-accessed`

The attempted operation is only legal on a deaccessed database.

`DBNOTOPEN 0BBC872A Error  Database not open`

An operation that is only legal on an open database has been attempted. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`DBNOTSTARTED 0BBC8532 Error  Database handler not started`

The database component has failed to start. Check the trace log for more information. Check the product installation.

`DBREQFAIL 0BBC830A Error  Database request failed`

The Add, Update, Delete or Lookup request failed because of a problem in the database server or database handler. Check the trace log for more information.

`DBSNOTLOC 0BBC8572 Error  Database server must be run locally`

The database server must run locally for the database recovery option. The product environment variable OMM_DB_SERVER must not be defined to "none".

`DBSQLERROR 0BBC874A Error  Database handler SQL execution error`

The database component has executed an illegal SQL call. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`DBTASKNOTCAN 0BBC8622 Error  Task not canceled`

A database task cancellation was not performed, probably because the task had already completed.

`DBTASKNOTSUSP 0BBC861A Error  Task not suspended`

Either a database suspend operation was not performed, probably because the task had already completed, or a resume operation failed because the task was not suspended.

`DEVALRRES 0BBC841A Error  Context already has a reserved device`

The context already has a reserved device, so another device cannot be reserved in it.

`DEVERR 0BBC80B2 Error  Device error`

An error has occurred on a device. Check the trace log for more information.

`DEVLOADEDERR 0BBC88FA Error  Operation failed because device has a tape loaded`

Some operations, such as deletion, can only be carried out on devices which do not have a tape loaded. Unload the tape and repeat the operation.

`DEVNOTCLR 0BBC8892 Error  Device is not clear`

A device is not available for an operation because there is already a tape loaded on it.

`DEVREQFORANO 0BBC818B Info  Device required for another client`

The admed server has generated a tape unload to free a device, to satisfy another client request. (This only happens if a previous client forgets to unload a tape when it has finished with it.)

`DEVRESANO 0BBC804A Error  Device is reserved by another client`

The device(s) referenced is/are already reserved by another client.

`DEVRESERR 0BBC88EA Error  Operation failed because device is reserved`

Some operations, such as deletion, can only be carried out on devices that are not reserved. Free the device and repeat the operation.

`DISCON 0BBC82B2 Error  Request not processed - connection is being closed`

This status is returned when a client sends a request on a connection that is being closed, usually because the service or admed server is being shut down.

`DUPCLIENT 0BBC86FA Error  Duplicate client identifier`

Each time a client connects to an admed server, it supplies a unique client identifier. This message indicates the same identifier was supplied twice to the same server. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`DUPSERVICE 0BBC84F4 Fatal  Service with the same attributes is already registered`

When the Admed server started, it found a service with the same attributes already registered with the service broker. This indicates that an Admed server is already running serving the same neighborhood.

`EJECTFAIL 0BBC889A Error  Tape eject failed`

An eject tape operation failed. The tape unload has been attempted, but may also fail.

`EMBSELMNY 0BBC84BA Error  Embedded selection returned more than one object`

The embedded selection criteria (@object field.op.value...) returned more than one match. In this case, the criteria must identify a unique object.

`EMBSELNON 0BBC84B2 Error  Embedded selection returned no objects`

The embedded selection criteria (@object field.op.value...) returned no matches but one (or possibly more) is required.

`EMPNEIANC 0BBC886A Error  Empty neighborhood field in anchor`

No neighborhood value has been set in the anchor API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`ERRCOMSRV 0BBC85F2 Error  Error communicating with server`

A communications error occurred when sending information to or receiving information from an admed server.

`ERRHSTINF 0BBC88DA Error  Failed to get host networking information`

A *gethostbyname* call failed. This is likely to be caused by a problem in networking data (as stored in /etc/hosts, for example).

`ERRHSTNAM 0BBC84D2 Error  Error getting host name`

An error was returned from a system *gethostname()* call. Client/server communications require the local host name. This indicates a problem with the platform socket package.

`ERROR 0BBC801A Error  Error`

This is a general error message. When it appears, there should also be other messages giving further details of the problem.

`ERRREDCFG 0BBC84DA Error  Error reading local config file`

An error occurred when trying to open or read the local configuration file *local_config.dat*. Check the trace log for more information. Check that the configuration file exists in the product data directory and has the correct access permissions.

`ERRREGSVC 0BBC84CA Error  Error registering service with service broker`

An error occurred while a server was trying to register its service with the service broker. Check the trace log for more information. Check that the LOCALBROKER entry in the local configuration file specifies the host that runs the service broker.

`ERRWRTCFG 0BBC84E2 Error  Error writing local config file`

An error occurred when trying to open or write the local configuration file *local_config.dat*. Check the trace log for more information. Check that the configuration file exists in the product data directory and has the correct access permissions.

`FILOPNERR 0BBC859A Error  Error opening file`

The system failed to open a file. Check the trace log for more information.

`FLDNOTINOBJ 0BBC87CA Error  Field is not a member of object`

When an operation requires an object and a field, the field must be a field located in that object. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`HOMEUNDEF 0BBC8522 Error  Cannot locate product home directory - environment variable OMMHOME not defined`

The environment variable OMMHOME is required to locate the product files. See your system manager to find out how to define OMMHOME.

`IDENTIFY 0BBC8702 Error  The anchorUuid field in the anchor does not contain a uuid`

The *anchorUuid* field must contain a uuid to uniquely identify the anchor. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`IMPTAPCHG 0BBC8922 Error  Impermissible change to reserved tape`

A tape that was reserved has experienced an unexpected or unanticipated external change.

`INPFILINV 0BBC843A Error  Input file to add/set has incorrect syntax at line __`

There is an error in the specified line in the file used as input to an ADD or SET command.

`INSPRGMEM 0BBC85A2 Error  Insufficient program memory`

An error occurred when trying to allocate program memory or expand an internal table. Check the trace log for more information. If the problem seems to be lack of resources, allocate more virtual memory to the program and restart.

`INTDELADD 0BBC82C2 Error  Delete and then re-add an intersection object to update it`

An intersection object (the expression of a many-many relation) cannot be updated. To make changes to an intersection object, delete it from the database then add it again with the changed fields. If using the CLI, use the ASSIGN and DEASSIGN commands.

`INTERR 0BBC865A Error  Internal error`

This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`INTERRCUI 0BBC8664 Fatal  Internal error in facility CUI`

An internal error has occurred in the Command User Interface. Note the operation in progress, save any trace information, and submit a fault report.

`INTERRZLC 0BBC866C Fatal  Internal error in facility ZLC`

An internal error has occurred in management of the local configuration file. Note the operation in progress, save any trace information, and submit a fault report.

`INVALCRED 0BBC8272 Error  Invalid credentials`

API: The credentials you supplied are not valid. Otherwise, this is an internal error: note the operation in progress, save any trace information, and submit a fault report.

`INVCMDOPT 0BBC8402 Error  Invalid command options`

The combination of command options and/or parameters is not valid.

`INVCMDPRM 0BBC8552 Error  Invalid command parameter`

An invalid command parameter was passed to a server.

`INVDOMAIN 0BBC827A Error  Invalid domain in credentials`

API: The credentials you supplied are not valid. Otherwise, this is an internal error: note the operation in progress, save any trace information, and submit a fault report.

`INVFLDNAM 0BBC834A Error  Invalid field name '__'`

The specified field name is not valid.

`INVKEYNAM 0BBC842A Error  Invalid keyword '__' for '__'`

The specified enumerated name is not valid.

`INVNUMBER 0BBC8352 Error  Invalid number`

A number was expected but not found.

`INVOBJNAM 0BBC8342 Error  Invalid object name '__'`

The specified object name is not valid.

`INVPASSWD 0BBC8262 Error  Invalid password`

The password supplied in a login attempt does not exist.

`INVREQCMD 0BBC880A Error  Invalid server instruction specified in requestInstructServer`

Several instructions to the admed server, e.g. *GetStatus*, are passed in a *requestInstructServer* structure. This error indicates that the admed server received a *requestInstructServer* containing an unrecognized instruction. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`INVREQREQ 0BBC8812 Error  Invalid request instruction specified in requestInstructRequest`

Several operations on requests, e.g. *GetStatus, Cancel*, are passed in a *requestInstructRequest* structure. This error indicates the admed server received a *requestInstructRequest* containing an unrecognized instruction. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`INVTIMSYN 0BBC84A2 Error  Invalid time syntax`

The textual form of a deltatime, timestamp or combination time is syntactically incorrect.

`JBXNODROP 0BBC8912 Error  Jukebox media location has no dropped media location`

The RECONCILE JUKEBOX command has been issued for a media location in a jukebox, but that media location has no dropped media location associated with it. A dropped media location is mandatory for a jukebox, as tapes moved out of the jukebox go to this media location.

`JBXNOTAVAIL 0BBC80A2 Error  Jukebox not available`

The jukebox is not available; it may be switched off.

`JBXREQMLC 0BBC88D2 Error  Jukebox has many medialocations, must specify one of them`

A RECONCILE JUKEBOX command has been issued for a jukebox with multiple media locations. One of these must be specified as the main media location.

`KEPT_CONTEXT 0BBC828B Info  Broken connection successfully reestablished`

This is an internal status exchanged between client and server to indicate that a broken connection has been reestablished, and the context of the connection (e.g. device reservations and change registrations) has been retained.

`LIMEXCEED 0BBC8302 Error  Specified matches limit exceeded`

The number of matches exceeded the LIMIT option. If the operation was one in which the CLI first looks up the object (e.g. delete or set), the operation is not performed on any of the matching objects. Use the CLI 'mode limit' command to set the limit.

`LIMITAPPLIED 0BBC82D3 Info  Request limit applied`

An operation has terminated because a caller-specified limit has been applied, rather than because the task is complete. For example, a lookup only returns objects up to the maximum number specified in the request, regardless of how many there are in the database.

`LOADFAIL 0BBC8112 Error  Load has failed`

A tape load operation has failed.

`LOADTIMEOUT 0BBC811A Error  Load has timed-out`

The load manager canceled an operation because it did not complete within a specified period.

`LOCNOTFND 0BBC855A Error  Found an object whose parent neighborhood is not the admedserver's neighborhood`

All objects should be related to a single neighborhood. This message indicates that an object has been found that relates to some other locality. Check the integrity of the database.

`LOSCONSRV 0BBC85FA Error  Lost connection to server`

A communication error occurred and the client lost connection during a transmission. The command may or may not have completed.

`LOST_CONTEXT 0BBC8290 Warning  Connection reestablished but context lost`

This status indicates that a broken connection has been reestablished, but the context of the connection (e.g. device reservations and change registrations) has been lost.

`MISANCPAR 0BBC8842 Error  Missing anchor parameter`

The anchor parameter in the call to the function is NULL. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`MISCONANC 0BBC8872 Error  No connection callback in anchor`

No connection callback has been set in the anchor. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`MISOBJNAM 0BBC8362 Error  Missing object name (last item scanned was '__')`

Syntax error: an expected object name was not found.

`MISREQFLD 0BBC882A Error  Missing required field`

An object sent to the admed server is missing one or more mandatory fields. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`MISREQPAR 0BBC884A Error  Missing request parameter`

The request parameter in the call to the function is NULL. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`MISRESCAL 0BBC8852 Error  Missing response callback`

No response callback has been supplied to the function. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`MISSTAPEINFO 0BBC80E2 Error  Tape information is missing`

The client has not supplied all the required information about a tape.

`MTXDBLGET 0BBC86AA Error  Handler of requestBlock __ (__) called zmutexGain when holding the mutex`

This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`MTXEXTCAL 0BBC88AA Error  Request attempted external wait while holding the mutex`

This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`MTXLOSERR 0BBC86B2 Error  Handler of requestBlock __ (__) called zmutexLose when not holding the mutex`

This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`MTXSTLHLD 0BBC86A2 Error  Handler of requestBlock __ (__) completed without dropping the mutex`

This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`MUSTACCRED 0BBC826A Error  Must accredit first, or supply credentials when connecting`

User interface: a previous login attempt has failed. API: Your first request must be an accredit request, or you must place a set of valid credentials in the anchor.

`MUSTBEUNIQUE 0BBC82EA Error  Field uniqueness violation`

A database operation cannot be completed because a particular field or combination of fields in the object type must be unique in the database, and the value supplied already occurs in another object in the database. For example, most objects have a Name field, which must be unique. Adding a host named 'europa' when a host named 'europa' already exists in the database would cause this error.

`MUSTCONNECT 0BBC86F2 Error  Must make a connect request first`

This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`NAMATTMIS 0BBC867A Error  Name attribute missing from service`

A service registered with the service broker does not have a name attribute. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`NAMTOOLNG 0BBC8442 Error  The specified name is too long`

The name has exceeded a system limit, and therefore cannot be valid.

`NEW_LOCALITY 0BBC8298 Warning  Reconnected to server in different locality`

This is an internal status exchanged between client and server. It indicates that the client, in trying to reestablish a broken connection, connected to an admed server managing a different locality from that of the original server. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`NMTNITPT 0BBC88C2 Error  New media type not in tape's physical type`

The proposed new media type does not reference the tape's physical type in a *NIRT* request.

`NOCOMMTRN 0BBC84EA Error  No communications transports (protocols) available`

No transport (e.g. tcp/ip) is available to support client/server communications. Check the PROTOCOL entry in the local configuration file. Check that the protocol stack is set up correctly.

`NOREGSTRD 0BBC80AA Error  No registered operators`

No operators are registered to perform a tape/device operation.

`NOSUCHACC 0BBC824A Error  Native user account does not exist`

The native user/host account does not exist. If you do not explicitly login, the user interface attempts a login based on your host and native username. This has failed because there is no UserAccount object for that combination.

`NOSUCHDEV 0BBC806A Error  No such device`

The supplied criteria or uuid does not reference a device known to the admed server.

`NOSUCHJBX 0BBC8082 Error  No such jukebox`

The referenced Jukebox object does not exist.

`NOSUCHMEDLOC 0BBC807A Error  No such media location`

The referenced MediaLocation object does not exist.

`NOSUCHMEDPOL 0BBC8072 Error  No such media pool`

The referenced MediaPool object does not exist.

`NOSUCHMEDTYP 0BBC808A Error  No such media type`

The referenced MediaType object does not exist.

`NOSUCHOPNSAVSET 0BBC8202 Error  No such open saveset`

The specified open saveset does not exist.

`NOSUCHOPNSEGMNT 0BBC8232 Error  No such open segment`

The specified open segment does not exist.

`NOSUCHPHYSTYP 0BBC8092 Error  No such physical tape type`

The referenced PhysicalType object does not exist.

`NOSUCHRECORD 0BBC82DA Error  No such record`

A database operation referenced an object that does not exist.

`NOSUCHREQ 0BBC82CA Error  No such request in server`

A request was issued to an admed server, either to cancel another request or get status on that request, and the specified request could not be located in the admed server. The request may have already completed, or the wrong *requestId* may have been supplied.

`NOSUCHRESDEV 0BBC805A Error  No such reserved device`

The ReservedDevice object referenced by its uuid does not match any such object held by the admed server.

`NOSUCHRESTAP 0BBC8052 Error  No such reserved tape`

The ReservedTape object referenced by its uuid does not match any such object held by the admed server.

`NOSUCHSAVSET 0BBC81E2 Error  No such saveset`

The referenced saveset does not exist.

`NOSUCHTAPE 0BBC8062 Error  No such tape`

No tape matches the supplied criteria, or the referenced tape does not exist.

`NOSUCHTAPORDEV 0BBC80FA Error  No such tape or device`

The specified tape/device combination does not exist or is not available.

`NOSUCHTORST 0BBC890A Error  No such tape or suitable scratch tape`

An available tape matching the client's criteria cannot be found, even when associated scratch pools are searched.

`NOSUCHUSR 0BBC825A Error  No such user`

The user name supplied in a login attempt does not exist.

`NOTLOADED 0BBC810A Error  Not loaded`

The device does not have a tape loaded on it so the specified operation is not valid.

`NOTONEDST 0BBC8132 Error  Not one destination (__ matches)`

The destination pool or location is ambiguous: more than one pool or location matches the specified criteria.

`NOTRECOVER 0BBC8323 Info  Database recovery not actioned`

The database recovery option was aborted.

`NOTREQUEST 0BBC881A Error  Object is not a request`

The object passed to *ExecuteRequest* is not a request. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`NOTSTARTED 0BBC86C2 Error  Manager not started`

This error is returned by the shutdown procedures of various managers when they are not started. It may appear during admed server shutdown if the server did not start correctly. Otherwise, it indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`NOTYETEXP 0BBC81F2 Error  Saveset has not yet expired`

The target saveset cannot be removed because it has not yet expired.

`OBJLOCKED 0BBC8642 Error  Operation failed - object is locked`

Objects are locked when the admed API needs to ensure that the object remains in existence and the values it contains do not change. The operations Init, Destroy, and Stencil are not allowed on a locked object. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`PARAMMIS 0BBC8612 Error  Parameter missing`

This message is returned by a number of API functions to indicate that a required parameter is missing. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

```
PLSREADLABEL 0BBC8103 Info  Please read label
```

The client program should read the label of the tape on the specified drive and send it to the admed server using the *ConfirmLoadTape* request. The request asking for this information may stall until the label is supplied.

```
PMTALRREG 0BBC844A Error  Already registered for prompts
```

In the CLI, you cannot register for prompts more than once.

```
PMTNOTREG 0BBC8452 Error  Not registered for prompts
```

You cannot deregister for prompts if you are not currently registered.

```
PRIREQTYP 0BBC87AA Error  Type of request is private
```

The client can not execute this type of request, as it is not public. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

```
RDSNOTLIC 0BBC86CA Error  RDS is not licensed on this platform
```

This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

```
REQINVDDB 0BBC849A Error  Request invalid for direct database connection
```

A request that requires a full admed server has been attempted while using a direct database connection (LOGIN/DATABASE). Log in to a full server.

```
REQMGRPRV 0BBC8582 Error  Operation requires management privilege
```

The attempted operation requires management privilege, but the caller does not currently have this privilege.

```
REQMISOBJ 0BBC87A2 Error  Request missing required object
```

The request is incomplete. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

```
REQNOTAVAIL 0BBC86DA Error  Request not available
```

The request is not currently implemented. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

```
REQNOTRDY 0BBC8122 Error  Request is not ready
```

A request is not in a state where it can accept the attempted intervention.

```
REQOPRPRV 0BBC857A Error  Operation requires operator or management
privilege
```

The attempted operation requires operator or management privilege, but the caller does not currently have either of these privileges.

```
SAVESETEMPTY 0BBC8213 Info  Saveset has no segments and will be deleted
```

The target saveset has no segments. *CloseOpenedSaveset* will delete the saveset.

```
SAVESETNOTOPEN 0BBC823A Error  Saveset is not open
```

The specified saveset cannot be closed because it is not open.

```
SAVESETOPEN 0BBC81EA Error  Saveset is open and must be closed first
```

The requested operation cannot be performed because the target saveset is open.

`SAVNOTOPN 0BBC8482 Error  No saveset open, cannot open a segment`

There is no saveset open on the specified context, so a segment cannot be opened on it.

`SAVSETEXISTS 0BBC81FA Error  Saveset already exists`

The target saveset cannot be created because a saveset with the same uuid already exists.

`SEGNOTCLOSED 0BBC820A Error  Saveset has open segments that must be closed first`

The target saveset cannot be closed because it has open segments.

`SELBADMANIP 0BBC8822 Error  Invalid manipulation for type of field`

The manipulation mask in a selection criteria relation is not legal for the type of field, or conflicting manipulations have been requested. For example, the field is not textual but upper case manipulation is requested, or upper and lower case manipulation are requested together. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`SELBADOP 0BBC87DA Error  Invalid operator in selection criteria relation`

The operator in a selection criteria relation is not legal. Legal operators are enumerated in the crit public header. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`SELBADOPUSE 0BBC87E2 Error  Invalid operator for type of field or value`

The operator in a selection criteria relation is not legal for the type of field or value. For example, the *like* operator used for wild card matching can only be used with textual fields, or only the *eq* and *ne* operators can be used in many-many lookups, lookups involving empty timestamps and lookups involving uuids, bools and other discrete valued fields. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`SELBADVAL 0BBC87EA Error  Invalid value in selection criteria relation`

The value in a selection criteria relation is not legal for the type of field. For example, the field is textual but the comparison value is numeric, or a timestamp comparitor contains illegal characters, or a non-uuid comparitor is used in a many-many lookup. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`SELNOTCRIT 0BBC87BA Error  Selections list or its members are not criteria relations`

The zlist in a criteria object has the wrong default object type or at least one of its elements has the wrong object type. Both should be criteria relation objects. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`SERVICESTOP 0BBC82AA Error  Request not processed - that service is stopping`

This status is returned to a client that sends a request to a service offered by the admed server when that service is being stopped.

`SETDERFLD 0BBC8432 Error  Cannot set derived name field '__'`

Derived fields (*NameOf...*) cannot be changed: they are generated automatically from the *UuidOf...* field.

`SETUNSUPP 0BBC88CA Error  Set of enum type to _UNSUPPORTED rejected`

You cannot set an enum field to _UNSUPPORTED. This value is no longer legitimate.

`SOCKETERR 0BBC852A Error  Failed to start sockets package`

During initialization, the platform socket package (e.g. Winsock on Windows NT) failed to start.

`SORTNOTOFOBJ 0BBC87FA Error  Sort number is not appropriate for object`

When objects are sorted, the sort must be appropriate for the object. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`SPAWNERR 0BBC8592 Error  Error spawning task`

Failed to spawn a subprocess or detached process Check the trace log for more information.

`STALECRED 0BBC8282 Error  Credentials are stale`

Credentials expire and may no longer be reused after 24 hours. API: Reaccredit rather than reusing existing credentials. Otherwise, this is an internal error: note the operation in progress, save any trace information, and submit a fault report.

`STASNDERR 0BBC8602 Error  Error sending statement`

A communication error has occurred when sending a request or response to another component. Check that the service broker is running correctly. Check the trace log for more information.

`STOPPING 0BBC82A2 Error  Request not processed - server is shutting down`

This status is returned to a client that sends a request to the admed server when the admed server is shutting down.

`SUCCESS 0BBC8009 Success  Successful completion`

This is a general success status.

`SUSPEND 0BBC8032 Error  Suspend operation`

This status should be returned from a *NoteProc* by an API client: it directs the component that called the *NoteProc* to suspend the operation.

`SVRDBHFIT 0BBC8502 Error  Server/db-handler Field Information Table mismatch`

Server and database statement handler have different versions of the table describing database object shapes. This message indicates a problem with the product installation.

`SVRDBHPRO 0BBC84FA Error  Server/db-handler communication protocol mismatch`

The server and database statement handler are running different versions of the protocol (request shapes) with which they communicate. This message indicates a problem with the product installation.

`SYNAMBCMD 0BBC83DA Error  Ambiguous command verb (last item scanned was '__')`

The command verb was ambiguous.

`SYNAMBKEY 0BBC83EA Error  Ambiguous keyword (last item scanned was '__')`

A keyword was ambiguous.

`SYNBADNUM 0BBC835A Error  Invalid number (last item scanned was '__')`

Unable to parse a number.

`SYNBADRNG 0BBC84AA Error  Bad range specification`

The range specification is invalid. Low and high value strings must be equal in length and use 0-9, A-Z or a-z for incrementable character positions.

`SYNFILINV 0BBC8382 Error  Syntax of filter is invalid (last item scanned was '__')`

Syntax error: the filter (criteria) was not valid.

`SYNINTERR 0BBC8792 Error  Syntax parser had internal error (last item scanned was '__')`

The parser failed an internal consistency check. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`SYNINVCMD 0BBC83D2 Error  Invalid command verb (last item scanned was '__')`

The command verb was not recognized.

`SYNINVCRI 0BBC83AA Error  Invalid criteria (last item scanned was '__')`

The criteria for '@' specification could not be parsed.

`SYNINVKEY 0BBC83E2 Error  Invalid keyword (last item scanned was '__')`

A keyword was not recognized.

`SYNINVOPR 0BBC837A Error  Invalid operator (last item scanned was '__')`

Syntax error: the operator was not valid.

`SYNINVOPT 0BBC8392 Error  Invalid option (last item scanned was '__')`

Syntax error: an option was not valid for this verb.

`SYNINVVAL 0BBC83A2 Error  Unexpected value (last item scanned was '__')`

Syntax error: an unexpected value was found.

`SYNMISCMD 0BBC83CA Error  Missing command verb (last item scanned was '__')`

Syntax error: no command verb.

`SYNMISDOT 0BBC836A Error  Missing dot (last item scanned was '__')`

Syntax error: an expected dot was not found.

`SYNMISOPR 0BBC8372 Error  Missing operator (last item scanned was '__')`

Syntax error: an expected operator was not found.

`SYNMISOPT 0BBC838A Error  Missing option (last item scanned was '__')`

Syntax error: an expected option was not found.

`SYNMISPAR 0BBC83FA Error  Missing required parameters (last item scanned was '__')`

Certain parameters are required, but have not been parsed.

`SYNMISVAL 0BBC839A Error  Missing value (last item scanned was '__')`

Syntax error: an expected value was not found.

`SYNNOTEXP 0BBC83F2 Error  Not expected here (last item scanned was '__')`

Syntax error: the element is not expected here.

`SYNNOTINV 0BBC83BA Error  Not valid in values (last item scanned was '__')`

The item found is not valid within values.

`SYNONEVAL 0BBC83B2 Error  Only one value is allowed here (last item scanned was '__')`

A list was found where only a single value was expected.

`SYNTOOMNY 0BBC83C2 Error  Too many values or parameters (last item scanned was '__')`

The implementation limit on the number of values or parameters has been exceeded.

`SYSPOLMLT 0BBC81C2 Error  System policy MM_MAX_LOAD_TRIES exceeded`

The system policy MM_MAX_LOAD_TRIES specifies the number of times the software should try to load a tape. After this number of tries, the request fails.

`TAPALRRES 0BBC8412 Error  Context already has a reserved tape`

The context already has a reserved tape, so another tape cannot be reserved in it.

`TAPDUEONS 0BBC816B Info  Tape has been moved offsite when due onsite`

The tape has been moved offsite when it is due to move onsite.

`TAPENOTLOADED 0BBC821A Error  Tape is not loaded`

You cannot open a segment on the specified tape because the tape is not loaded.

`TAPEREADONLY 0BBC81DA Error  Cannot open segment on read only tape`

You cannot open a new segment on a tape because the tape has been loaded read only.

`TAPERESANO 0BBC8042 Error  Tape is reserved by another client`

The specified tape, or all tapes matching the criteria, is/are already reserved by another client.

`TAPERR 0BBC80BA Error  Tape error`

Error on tape. Check the trace log for more information.

`TAPISBAD BBC81A2 Error  Tape is bad`

The Tape.Bad field is true. This prevents the attempted operation.

`TAPISLOCKED 0BBC81AA Error  Tape is locked`

The Tape.Locked field is true. This prevents the attempted operation.

`TAPLOADEDERR 0BBC88F2 Error  Operation failed because tape is loaded`

Certain operations (e.g., deletion) can only be carried out on tapes that are not loaded. Unload the tape and repeat the operation.

`TAPLODDEV 0BBC8182 Error  Tape cannot be moved as it's currently loaded on a device`

The tape is loaded on a device so cannot be moved. Unload the tape and try again.

`TAPNEWLOC 0BBC8153 Info  Tape moved to a location that is not an onsite or offsite location`

The tape has moved to a location that is not an offsite or onsite location in the tape object.

`TAPNMDEV 0BBC809A Error  Tape is not mountable on device`

The tape and device reside in different jukeboxes or are otherwise incompatible.

`TAPNOTEXP 0BBC814A Error  Tape has not expired`

The tape has not yet expired: its *CompExpirationField* (computed expiration) is in the future.

`TAPNOTWRT 0BBC80F2 Error  Tape is not write-enabled`

A tape has been loaded write-protected when it was required write-enabled.

`TAPOFFEAR 0BBC8163 Info  Tape has been moved offsite too early`

The tape has moved offsite before it was due to move.

`TAPOPNSEG 0BBC81D2 Error  Tape has opened segments`

The requested operation cannot be performed because the tape has opened segments.

`TAPREQFORANO 0BBC8193 Info  Tape required for another client`

The admed server has generated a tape unload to free the tape for loading elsewhere.

`TAPRESERR 0BBC88E2 Error  Operation failed because tape is reserved`

Certain operations (e.g., deletion) can only be carried out on tapes that are not reserved. Free the tape and repeat the operation.

`TAPRETEAR 0BBC815B Info  Tape returned onsite too early`

The tape has been returned onsite before it was due.

`TAPSAMLOC 0BBC8173 Info  Tape is already at selected location`

The tape is already at the location noted by the request.

`TAPSTSBAD 0BBC819A Error  Tape status is wrong for this operation`

The tape has the wrong status for this operation.

`TIMEOUT 0BBC803A Error  Activity timed out`

This is a general error indicating that an activity did not complete within the time allowed for it.

`TOOMNYCTX 0BBC840A Error  Too many contexts`

Too many separate contexts are active.

`UNIDEVREQ 0BBC81BA Error  Unique device required`

Criteria identifying only one device are required. Too many devices match the criteria supplied.

`UNITAPREQ 0BBC81B2 Error  Unique tape required`

Criteria identifying only one tape are required. Too many tapes match the criteria supplied.

`UNKPROREP 0BBC8832 Error  Unknown reply type in prompt header`

It is not possible to raise the specified prompt because it doesn't specify the type of reply expected. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`UNKREPTYP 0BBC88A2 Error  Unknown reply type`

The load manager received a reply to a prompt, but the reply type is wrong. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`UNLOADFAIL 0BBC88B2 Error  Tape unload failed`

An unload tape operation has failed. This status is returned from the jukebox controller, or manually from the GUI or CLI.

`UNRPMTTYP 0BBC86D2 Error  Unrecognized prompt type (__)`

Cannot reply to the specified prompt, because its type is unknown. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`UUIDFMTERR 0BBC832A Error  Uuid format error`

The system failed to convert a uuid string into a uuid. The format should be "%8x%4x.%4x.%2x.%2x.%2x.%2x.%2x.%2x.%2x.%2x", for example 123456781234.1234.56.78.9a.bc.de.f0.12.34.

`VALTOOHI 0BBC8652 Error  Set failed - value too high`

The field is specifies a maximum value and a Set, Stencil or Make function tried to set it to a value higher than the maximum. API: Check your code. User interface: Check your input. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`VALTOOLOW 0BBC864A Error  Set failed - value too low`

The field is specifies a minimum value and a Set, Stencil or Make function tried to set it to a value lower than the minimum. API: Check your code. User interface: Check your input. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`WNTSCMERR 0BBC85AA Error  Error communicating with Service control manager`

The system failed to send an instruction to the Windows NT service control manager (for example to start a service). Check the trace log for more information.

`WORKING 0BBC8013 Info  Success so far; still working`

The request has been accepted for processing but is not yet complete.

`WROCONTYP 0BBC8862 Error  Wrong connection type`

ExecuteRequestAsynch has been called on a synchronous connection, or *ExecuteRequestSync* has been called on an asynchronous request. API: Check your code. Otherwise, this indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`WRONGTAPE 0BBC80C2 Error  Wrong tape`

A tape other than the one requested has been supplied.

`ZAACNTREP 0BBC863A Error  Cannot send reply for task __`

The jukebox controller could not complete the specified task because it could not send a reply to the admed server. Check the trace log for more information.

`ZAADEVMAN 0BBC854A Error  Device __ is managed by another jukebox controller`

The jukebox controller was asked to manage a device, but it is being managed by another controller. The controller has ignored the request to manage this device.

`ZAADUPDEV 0BBC8542 Error  Duplicate device parameter: __:__ __:__`

The jukebox controller has been told twice to manage the same device. Correct the commandline used to start the controller.

`ZAAEMTRSP 0BBC877A Error  Empty response`

The jukebox controller received a response to its prompt registration that contained no prompt. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`ZAAINVARG 0BBC85D2 Error  Invalid commandline argument __`

replytojbx or the jukebox controller have been passed an invalid commandline argument. They should be of the form *name*:*value* Refer to the sample jukebox_driver script.

`ZAAINVNAM 0BBC85DA Error  Invalid parameter name __`

replytojbx has been passed an invalid parameter name. Refer to the sample jukebox_driver script.

`ZAAINVOPR 0BBC876A Error  Invalid operation __`

A task spawned by the jukebox controller has been instructed to perform an operation it does not recognize. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`ZAAINVPRM 0BBC8772 Error  Invalid prompt type`

The jukebox controller did not recognize the type of prompt it received. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`ZAAINVRET 0BBC85EA Error  Replytojbx returned invalid name=value pair`

The jukebox controller received an unknown parameter name from a spawned task, or a value it did not recognize. Check the *replytojbx* call in the jukebox_driver script. Refer to the sample jukebox_driver script.

`ZAAINVTSK 0BBC8782 Error  Invalid taskid received from replytojbx`

The jukebox controller received an invalid response from a spawned task. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`ZAAMISARG 0BBC85E2 Error  Missing commandline argument(s)`

replytojbx is used to send name=value data pairs to the jukebox controller. Supply these on the commandline: replytojbx mediatype=tape. Refer to the sample jukebox_driver script. The jukebox controller expects one or more commandline arguments, either jukebox name, or jukebox:device.

`ZAAMISENV 0BBC8762 Error  Missing environment variable __`

A task spawned by the jukebox controller cannot find an environment variable. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`ZAAMISTSK 0BBC878A Error  Cannot find task`

The jukebox controller received a response from a spawned task but could not find the task on its internal list. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`ZAANODEVS 0BBC853A Error  No devices to manage`

The jukebox controller has no valid devices to manage: all those supplied either do not exist in the database or are being managed by another jukebox controller. Another jukebox controller may be managing the same device.

`ZAATSKERR 0BBC870A Error  Error getting spawned task exit code`

The jukebox controller failed to check the status of one of its spawned tasks. This is an internal error. Note the operation in progress, save any trace information, and submit a fault report.

`ZLMJBXUNK 0BBC81CA Error  Unknown jukebox class`

The load manager has received instructions relating to a jukebox whose class it does not recognize.

`ZPMNOTSNT 0BBC862B Info  Prompt raised but no client is currently registered for prompts`

The prompt has been raised but not (yet) sent to any client. This occurs if no client is currently registered for prompts, or if the clients that are registered do not wish to receive prompts from this source.

`ZPMNOTSTA 0BBC86BA Error  Prompt manager not started`

The prompt manager startup routine has not yet been called. This message may appear during admed server shutdown if the server did not start correctly. Otherwise, it indicates an internal error: note the operation in progress, save any trace information, and submit a fault report.

`ZPMPROMIS 0BBC8632 Error  Prompt not found`

The prompt was not found on the outstanding prompts list. The uuid of the prompt may be incorrect or the prompt may have already been answered.

`ZSDSTDERR 0BBC851A Error  Detached processes cannot send trace to standard error`

Product trace output has been directed to a standard error by defining the environment variable OMM_TRACEDIR to a nonexistent directory. This is not permitted for detached background processes.

Glossary

Accreditation

See Password accreditation and Native accreditation.

Administration object

An administration object is an object that represents an administrative item in the enterprise. Media pools, localities, users and hosts are examples of administration objects.

See also *Database object, Media object, Volatile object*.

Agent

The OpenMedia Agent starts and stops the OpenMedia service broker, server and jukebox controller software on a host. It uses the host's local configuration file to determine which (if any) of these are required when OpenMedia is started on the host.

API

The OpenMedia Application Program Interface (API) is available as an option to the OpenMedia package. It is a development tool that has been designed to help Value Added Developers (VADs) and Original Equipment Manufacturers (OEMs) integrate OpenMedia with their own products. OpenMedia client applications such as the GUI and CLI have been developed using the API. For more information about the API, see the manual *HYPERtape OpenMedia Application Program Interface*.

Application program interface

See API.

Backup Node

The HYPERtape Backup Node stores the backup data as savesets. The Backup Node host should be a system that provides automated tape storage using a tape robotic library and a tape management system. HYPERtape supports more than one Backup Node on the network.

Backup object

A HYPERtape backup object is a record in the HYPERtape object database, which is held on the HYPERtape Control Node. During a backup operation, the Control Node passes information about the relevant backup object to the Backup Node. This includes information relevant to OpenMedia, such as the saveset name, media information and a number of flags.

Bool data type

A bool data type represents a field with two possible states. In the CLI, a bool data type can take the values YES or NO. In the GUI, a bool data type takes the form of a checkbox.

CLI

The CLI is a client application that provides a way of issuing server requests directly from the command line. Its primary purpose is to allow the development of scripts for managing media and/or integrating OpenMedia with other applications. It can also be useful for remote support operations over dial up modems with connections such as telnet.

Client

OpenMedia uses a client/server architecture. Client applications connect to a server and issue requests such as to reserve and load tapes on devices. The server executes requests issued by clients and updates the object database. Examples of client applications are the GUI and the CLI.

Command Line Interface

See CLI.

Connection

A connection is a volatile object which represents the connection between a client and a server. Current connections can be viewed via the GUI, or by issuing the CLI command SHOW ANCHOR.

Control Node

The HYPERtape Control Node performs all the database operations for HYPERtape backup objects and initiates the backup job. It should be a network node with maximum up time which has access to all Service Nodes on the network. Account and History databases are kept on the Control Node.

Database object

A database object is an OpenMedia object that is maintained in the OpenMedia database.

See also *Volatile object, Administration object, Media object.*

Deltatime data type

Deltatime fields hold intervals of time. Examples of deltatime fields are the default onsite and offsite durations in a media pool object.

Device object

A device object is a database object representing a device such as a tape drive. Device objects hold details of the device's capability in terms of the media types it can read and write, the hosts that can access it and whether it is part of a robotic jukebox. Usage statistics are maintained for each device.

Device status

A device status is a volatile object created by the server to represent the current status of a reserved device or a device with a tape currently loaded on it. Device statuses are treated as extensions to devices and reserved devices while the device is in use.

Drag and drop

Drag and drop is the technique of using the mouse to move the name of an object in a GUI browser window to a field in a GUI detail window. This technique is particularly important with the OpenMedia GUI, as you use it when you create links between database objects. The implementation of drag and drop is specific to OpenMedia.

Enum data type

The value of an enum data type is a keyword. In the GUI, the available keywords are shown in a pull down list. When using the CLI, you can issue the DISPLAY command to show the available keywords.

Graphical User Interface

See GUI.

GUI

The OpenMedia Graphical User Interface (GUI) is a client application that allows manipulation of objects in the database from a host workstation. For example, you can interrogate the database to discover which tapes are past their expiration dates, then scratch them ready for reuse. The purpose of the GUI is to allow the System Manager to manage, interrogate and change data in the OpenMedia database.

Host object

A host object is a database object that represents a host system in the enterprise. A host object is required for any host that has tape devices or that controls devices or jukeboxes. You do not need a host object to run the GUI or the CLI, but failure to create on prevents the local configuration file from being updated automatically and causes errors if the OpenMedia service is started on the host.

Jukebox object

A jukebox object represents either a collection of manual tape devices controlled by the same (human) operator, or an automated tape library with devices controlled by a robot arm.

Local configuration file

The local configuration file, LOCAL_CONFIG.DAT, resides in the data subdirectory of the OpenMedia home directory of each host on which OpenMedia is installed. The contains information that allows the startup agent to start up the OpenMedia system and to enable a client application such as the GUI or CLI to connect to the OpenMedia server.

When OpenMedia starts on the host, the startup agent uses the file to determine whether the host runs server software, the network protocol, the identity of the host's neighborhood locality and the names of one or more service broker hosts for that locality. When the GUI and CLI are started, they attempt to connect to the neighborhood locality defined in the local configuration file.

Locality object

A locality object represents a site within the enterprise. Each locality is served by a single server.

Media location object

Media locations are used to determine whether tapes can be loaded into devices. A jukebox may contain several media locations, particularly if it is an operator class jukebox representing the manual tape drives that an operator can load. However, a media location may also exist outside any jukebox. A tape is loadable on a device if the tape resides in a media location which is itself in the same jukebox as the device.

Media object

A media object is an OpenMedia object that represents a media item in the enterprise. Tapes, devices and jukeboxes are examples of media objects.

See also *Administration object, Database object, Volatile object*.

Media pool object

A media pool is a site defined grouping for tapes. It allows a group of tapes to be treated identically with respect to their movement between locations and the retention of their data.

Media type object

A media type represents how data is recorded on a tape. Tapes have a media type unless they are blank, in which case they only have a physical type. Devices have a list of media types that they can read and a list of media types that they can write.

Native accreditation

A method of accreditation used when logging in to OpenMedia. When you use native accreditation, the server uses a combination of the host name and native username of the client system to find and check your OpenMedia user name. In order to use native accreditation, the OpenMedia database must contain a user account object for the host and native username.

Other HYPERtape components such as the Backup Nodes use native accreditation when logging in to OpenMedia.

See also *Password accreditation*.

Notification box

If OpenMedia detects an error, it displays a *notification box* giving some explanation of the problem.

For a full list of OpenMedia messages with explanations, see Appendix B.

Object database

OpenMedia maintains a database of objects, which can be physical items such as the tapes and host systems in your enterprise, or abstract entities such as media pools and user privileges. The objects and the defined relationships between them are encompassed in the OpenMedia model.

Opened saveset

An opened saveset is a volatile object created by the server to represent a saveset when it is being committed to tape.

OpenMedia service

The OpenMedia service on a particular host consists of the components that are started by the OpenMedia Agent. These include the server, service broker and jukebox controllers.

Password accreditation

A method of accreditation used when logging in to OpenMedia. If you use password accreditation, you must supply a valid OpenMedia user name and password as well as the neighborhood locality of the server with which to connect.

See also *Native accreditation*.

Physical type object

A physical type represents a particular type of tape, which may then be written in several different media types. For example, a DLT Tape IV can be written at several different recording densities, depending on the drive and options used to write the tape. The physical type governs whether the tape can be loaded into a drive and is mainly used when a tape is blank, to determine if a drive can initialize it.

Private media pool

A private media pool is the simplest type of media pool. It is a set of tapes reserved for a particular usage pattern and is normally used when a special set of tapes is bought for a particular job. Tapes move automatically into a private pool from its associated scratch pool, but they do not move automatically out of a private pool: you move the tapes manually. The tapes in a private media pool typically cycle through the usage states of blank, scratch, normal and full.

Privilege object

Privileges are required to do media management. There are two privileges: Manager and Operator. Operator privilege allows the user to answer prompts. Manager privilege is required to manipulate tapes and devices.

Public media pool

A public media pool draws tapes from an associated scratch pool when it needs new scratch tapes, and returns tapes to the associated scratch pool when all the data on them has been deleted or expired and they are scratched. This means that the scratch tapes are pooled between all jobs and so uses far fewer tapes.

Related object

Multiple relationships between two types of object are stored in the database as lists of related objects. They appear in the Related objects tabs of the GUI detail windows.

Reserved device

A reserved device is a volatile object created by the server to represent an exclusive reservation of a device for manipulation by the client. A reserved tape can be loaded on a reserved device and have a saveset opened on it.

Reserved tape

A reserved tape is a volatile object created by the server to represent an exclusive reservation of a tape for manipulation by the client. A reserved tape can be loaded on a reserved device and have a saveset opened on it.

Retention

The media pool retention is the default period for which data is written to tapes in the media pool. If the application writing a saveset to a tape does not set the saveset's expiration date, the media pool's retention is used to calculate an expiration date for the saveset.

Robot Manager

The HYPERtape Robot Manager provides the link between OpenMedia and the robot devices. From any location, you can load and unload tapes, move tapes from any source to any destination, load tapes using a bar code label or slot number and view the contents and status of all robots.

Because of the Robot Manager's client/server architecture, you can view and manage robots throughout the network from a central point. This allows you to deploy remote storage robots and so optimize your network backup throughput.

Saveset object

A saveset object represents a saveset held on one or more tapes. You should not normally modify saveset objects. They are updated automatically by OpenMedia or client applications.

Scratch media pool

Scratch media pools contain scratch and blank tapes. They are all available for use and move to other media pools when needed. With the aid of the OpenMedia facilities for database interrogation and scratching tapes, you can use scratch media pools to maintain an adequate inventory of reusable tapes. This is often the most economical way of using tapes as you do not need to keep spare tapes in other media pools.

Segment object

A segment object represents the portion of a saveset that resides on one tape. You should not normally modify a segment object. Updates are carried out automatically by OpenMedia and the client application that uses the segment. Segments are automatically removed by OpenMedia when the tape containing them is scratched.

Server

OpenMedia uses a client/server architecture. Client applications connect to a server and issue requests such as to reserve and load tapes on devices. The server executes requests issued by clients and returns one or more responses.

Service broker

The OpenMedia service broker connects a client application to a service. OpenMedia uses *dynamic* service broker lookup, whereby clients ask for a service rather than a particular server and are connected to any server offering that service. This means that only the service brokers need to know where items such as the server and jukebox controllers reside, so individual clients are not affected if such items are moved to new hosts. You can have a number of service brokers in the network, to provide resilience in the event of a failure and to avoid any client reconfiguration.

Service Node

A HYPERtape Service Node is a holds data that is backed up in a HYPERtape backup operation and performs that operation. A Journal database is kept on the Service Node for the objects saved from the node.

System policy object

System policy objects are used to set certain system defaults, such as the maximum number of times OpenMedia will attempt to load a tape and the device allocation method to be used. For details of these objects, see Chapter 2. Default values are set for all these objects during the installation procedure.

Tape object

A tape object represents a tape in the enterprise. You should not normally need to modify tape objects unless there are problems with the tape and you want to mark it bad, or you want to add or change a comment. All other information is automatically updated by OpenMedia or the client application that uses the tape.

Tape status

A tape status is a volatile object created by the server to represent the current status of a reserved tape or a tape currently loaded on a device. Tape statuses are treated as extensions to tapes and reserved tapes during these periods.

Timestamp data type

A timestamp represents an actual time. It takes the format yyyy-mm-dd hh:mm:ss.ffff. For example, 1997-10-13 22:15 is quarter past ten on 13th October, 1997.

Universally unique identifier

See UUID.

User account object

User accounts allow logins from specific hosts without the need of a password. A user account object specifies a host, a native user (the username used to log in to the host) and an OpenMedia user. The installation of the server on Windows NT creates user account objects for the Administrator and SYSTEM users from the server host.

User object

Before you can use any OpenMedia facilities, you need to log in to an OpenMedia user. User objects have privileges within a locality.

UUID

The universally unique identifier (uuid) is OpenMedia's internal mechanism for identifying and relating objects.

Uuid data type

A uuid data type is an internal reference to another object, represented by an alphanumeric string.

Volatile object

A volatile object is an OpenMedia object that is maintained in the server memory, and not in the OpenMedia database. Volatile objects represent transient states, such as the load status of tapes and devices. If these transient states were updated in database objects, they could cause inaccuracy or inconsistency in the database in the event of a system failure.

See also *Administration object, Database object, Media object*.

Index

A

Abbreviations 56
Abnormal termination of the server 142
Accessible by device 98
Accessible by host 116
Accreditation 142
ACSLS 24, 25
Activate button 34, 44
Activating windows 34
ADD 23, 24, 52, 61, 62
Add button 44
Add empty button 44
Add reference button 44, 48
add-browses command script 74
add-browses.txt 74
add-devices.txt 18
add-hosts.txt 16
Adding columns 40
Adding database objects 46
Adding related objects 47
Adding tapes to the database 24, 25, 26, 121
add-jukeboxes.txt 16
add-medialocations.txt 17
add-mediapools.txt 17
add-mediatypes.txt 16
add-physicaltypes.txt 16
add-systempolicies.txt 19
Administration object 12, 94
After image journal backup 90
Agent 12
ALL 69, 81
All prompts 127
API 12, 14, 141
Append flag 29, 30
Appending files to a tape 91
Application 133, 136
Application Program Interface *See* API
Architecture 12, 51
Arrived date 132
ASSIGN 52, 63, 82
ASSIGNS 78, 79
Associated scratch pool 95, 104
Autoclean by duration 113
Autoclean by errors 113
Autoclean by load duration 113
Autoclean by loads 113

B

Backing up the OpenMedia database 120
Backup frequency 90
Backup generations 90
Backup Node 20, 21, 23, 24, 27, 28, 31
Backup object 28
Backup retention 90
Backup strategy 87, 88, 89
Backup types 89
Backup window 88, 89, 90
Bad 131
Bad tapes 137
Bar code 131
Before starting the OpenMedia service for the first time 21
BLANK 24, 25, 26, 94, 131
Bool 58, 60
Brokered by host 96
Brokers for locality 98
Browse "All prompts" window 127
Browse "Devices" window 116
Browse "Hosts" window 97
Browse "Jukeboxes" window 112
Browse "Localities" window 96
Browse "Media locations" window 114
Browse "Media pools" window 102
Browse "Media types" window 100
Browse "Native host user accounts (proxies)" window 107

Browse "Physical media types" window 100
Browse "Privileges" window 108
Browse "Saveset segments" window 134
Browse "Savesets" window 135
Browse "System policies" window 105
Browse "Tapes" window 130
Browse "Users" window 106
Browse... button 41, 44, 47
Browser window 35, 36, 39
BTW 56
BY_USAGE 104, 122

C

Cancel button 44
Capacity 101
Changes 52, 67
Char value 105
CHECK 76
Checkbox 41
Class 113
CLEAN DEVICE 63
Clean duration 113
Clean unload retries 113
Clean unload retry delay 113
CLEANED 71
Cleaner 131
Clear button 41, 44
CLI 12, 13, 21, 22, 51
CLI architecture 51
CLI command categories 51
CLI commands 62
Client 141
Client absolute tape position 134
Client date 133, 136
Client number 133, 136
Client relative tape position 134
Client text 133, 136
CLOSE SAVESET 52, 64
CLOSE SEGMENT 52, 61, 64
Color 35, 50
Column 41
Combination dates and times 60
Command button 44
Command categories 51
Command limit 61
Command Line Interface *See* CLI
Commands 62
Commands that interrogate or manipulate database objects 51
Commands that make server requests 53
Commands that manage media objects 52
Commands that process prompts and changes 52
Comment 97, 98, 100, 101, 103, 105, 107, 113, 114, 117, 130
Comms protocol1 98
Comms protocol2 98
Comms protocol3 98
Computed expiration date 131
Configuring OpenMedia 23
Configuring the database 23
CONFIRM 66, 78
Confirm delays 117
Confirm retries 117
Connections 139
CONTEXT 63, 64, 71, 72, 73, 76, 77, 81
Control Node 28, 29, 30
COPY 23, 24, 52, 61, 65
Copying device objects 24
Cost and quantity of storage media 89
Counter field 41
Creation date 135
Cumulative incremental 89
Current date and time button 44
Customize menu 35
Customizing browsers 40

D

Daily processing 123
Data retention 86
Data type 58
DATABASE 69
Database backup 90
Database menu 35, 121
Database object 35, 46
Database server 21
database_maintenance 120
Date and time specification window 42
Date field 42
DB_MAX_HANDLERS 19
DEASSIGN 52, 66, 82
Default assignment style 61
Default offsite media location 104
Default onsite media location 104
Defaults 61
DELETE 52, 61, 66
Delete button 44, 48
Deleting columns 41
Deleting database objects 48
Deleting reports 74
Deleting tapes 121
Deltatime 42, 58, 59, 60
Deltatime specification window 42
DEREGISTER 53, 67
Detail window 37, 41
Developing a backup strategy 87
Developing a storage management strategy 85

Device 18, 24, 46, 111, 115, 116
Comment 117
Confirm delays 117
Confirm retries 117
Error count 118
Errors since last clean 118
Last cleaned date 118
Last loaded date 118
Load count 117
Load count since last clean 118
Load duration 118
Load duration since last clean 118
Locked 117
Name 116
Name of control host 117
Name of jukebox 117
Name of locality 117
Physical name 116
Unit number 117
Device and media management 86
Device by error 138
Device on host window 99, 119
Device pref by location window 115, 119
Device preferences 115
Device statistics 117
Device status 139
Device window 116
Devices accessible by hosts 19
Devices available for backing up 88
Devices selection window 115, 127
DIRTY 129, 131
DISCARD 64
Dismount flag 30
DISPLAY 52, 53, 55, 59, 67
Drag and drop technique 43
Dropped media location 25, 26
Due offsite date 132
Due onsite date 132
Dynamic data 88
Dynamic service broker lookup 12

E

ECHO 53, 67
Embedded selection criteria 57
EMERGENCY 78
END 53, 54, 68
Entering data into detail windows 41
Enum 58, 60
ENVIRONMENT 79
EQ 56
Error and disaster recovery 142
Error count 118, 133
Error log 34
ERRORCOUNT 73
Errors since last clean 118
EXIT 53, 54, 68, 72
Expand button 45
Expand Prompt button 128
Expiration date 136
EXPIRE SAVESET 52, 68, 125
Expire savesets 125
Expire savesets selection window 125
Expired savesets 138
Expired tapes 138
Expired tapes selection window 124
Expiring savesets 125
EXPORT 75
External label 130

F

FILE 62, 64, 65, 67, 72, 75, 76, 78, 80, 81
File menu 35
Finding your way round the GUI 36
First used date 133
FIXED 104
Font 35, 50
Font window 50
FOREIGN 25, 26, 131
FORMAT 71, 130
Frequency of backups 90
FULL 131
Full backup 89

G

GE 56
General administration 94
General syntax 54
General tab 37
GENERIC 16, 113
Getting help 45, 55
Graphical User Interface *See* GUI
GT 56
GUI 12, 13, 21, 22, 33, 34
GUI facilities 13

H

Handle 38
HEADER 62, 67, 75, 80
Help 45, 53, 55, 68
Help menu 35, 45
Host 16, 23, 46, 94, 97, 98, 108, 111
Comment 98
Comms protocol1 98
Comms protocol2 98
Comms protocol3 98
Name 98

Name of locality 98
Runs admed server 98
unique name 23, 98
Host brokering window 97, 99
Host window 98
Hosts selection window 97
Htape_init 24, 27
HTAPE1 27
HTAPE2 27
HTRM *See* HYPERtape Robot Manager
HYPERtape Backup Node *See* Backup Node
HYPERtape backup object 28
HYPERtape Control Node 28, 29, 30
HYPERtape Robot Manager 23, 24, 25
HYPERtape Robotic Management Interface 24, 25

I

Identifying tapes to be moved 122
Implementing your storage management strategy 91
Incomplete 135
Incremental backup 89
Index in stream 134
Index on tape 134
Initial considerations 93
INITIALIZED 71
Initializing tapes 24, 27
Initializing tapes for use with HYPERtape 27
Installation 15
Installing OpenMedia 15
Int value 105
Integrating with robot managers 25
Integrating with SCH:Robot 26
Internal label 130
Internal privilege 109
INTERNALLABEL 72
Interrogation options 137
Interrogations 35, 136
Interrogations menu 35
Introducing OpenMedia 11
Introducing the CLI 51
Is domain 97
Is neighborhood 97
Item lists 61
ITEMS 62, 64, 65, 72, 75, 78, 80

J

JBX_MAX_HANDLERS 19
JOB_DAILY_PROCESSING 19, 123
Jukebox 16, 24, 25, 26, 27, 29, 30, 46, 111, 112
Autoclean by duration 113
Autoclean by errors 113
Autoclean by load duration 113
Autoclean by loads 113
Class 113
Clean duration 113
Clean unload retries 113
Clean unload retry delay 113
Comment 113
Locked 113
Model 113
Name 112
Name of control host 113
Name of locality 113
Physical name 113
Jukebox controller 12, 21, 22, 23, 111
Jukebox integration script 121
Jukebox reconciliation 24, 27
Jukebox window 112
jukebox_driver 111
Jukebox_reconcile_*robottype* 25
Jukeboxes selection window 112

L

Last cleaned date 118
Last loaded date 118, 133
Last loaded for write 133
LE 56
Leaving the GUI 50
LIKE 56
LIMIT 61, 63, 64, 65, 66, 68, 70, 71, 73, 75, 77, 78, 80
Limitations affecting a backup strategy 88
LIST 69, 75
Listing database objects 46
LM_LOADTAPE_TIMEOUT 20
Load count 117, 133
Load count since last clean 118
Load duration 118
Load duration since last clean 118
Local configuration file 20, 21, 22, 23, 33, 69
LOCAL_CONFIG.DAT 20, 33, 69
LOCALADMIN 20
LOCALBROKER 20
LOCALHOOD 20
Localities selection window 96
Locality 15, 34, 46, 94, 96, 106, 111, 141
Comment 97
Is domain 97
Is neighborhood 97
Name 97
Name of locality 97
Timezone 97
Locality window 96
LOCATION 70
Lock 131
Locked 113, 117

Locked devices 138
Logging in 35
LOGIN 53, 54, 68
Login window 35
Look-and-feel 13, 43
Looking up connections 139
Looking up device statuses 139
Looking up opened savesets 139
Looking up reserved devices 138
Looking up reserved tapes 138
Looking up tape statuses 139
Lookup button 45
Lost connection to a client 142
Lost privileges 109
Lost users 109
LT 56

M

Mail address 107
Main window 34
MANAGER 16, 142
Manager privilege 109
Manager user 15, 21
Managing database objects 46
Managing media operations 111
Managing stored data 121
Managing the movement of tapes 122
Manipulating database media objects 111
Manipulating device objects 115
Manipulating host objects 97
Manipulating jukebox objects 111
Manipulating locality objects 96
Manipulating media location objects 114
Manipulating media pool objects 102
Manipulating media type objects 100
Manipulating physical type objects 99
Manipulating privilege objects 108
Manipulating saveset objects 135
Manipulating segment objects 134
Manipulating tape objects 129
Manipulating tapes, segments and savesets 129
Manipulating user account objects 107
Manipulating user objects 106
Mark devices cleaned 127
Mark devices serviced 127
Mark tapes offsite 126
Mark tapes onsite 126
Marking devices cleaned 127
Marking devices serviced 127
Marking tapes offsite 126
Marking tapes onsite 126
Max streams 136
MAXIMUM 75
Media location 17, 24, 25, 26, 29, 30, 46, 111, 114
  Comment 114
  Name 114
  Name of dropped media location 115
  Name of jukebox 115
  Name of locality 115
Media location window 114
Media locations selection window 114
Media management 86
Media object 12, 93, 111
Media pool 17, 28, 29, 30, 47, 94, 95, 102, 103, 129
  Comment 103
  Movement cycle 104
  Name 103
  Name of associated scratch pool 104
  Name of def offsite media location 104
  Name of def onsite media location 104
  Name of locality 103
  Offsite duration 104
  Onsite duration 104
  Overflow count 104
  Retention 104
  Type 103
  Underflow count 104
Media pool window 103
Media pools selection window 102
Media type 16, 27, 29, 30, 47, 94, 100, 101
  Capacity kb 101
  Comment 101
  Name 101
  Name of locality 101
  Name of physical type 101
Media type window 101
Media types associated with devices 18
Media types selection window 100
MEDIATYPE 72
Menu bar 34
Menu items 35
Message format 143
Message list 143
Miscellaneous commands 53
MM_DEFAULT_CONFIRM_DELAY 20
MM_DEFAULT_CONFIRM_RETRIES 20
MM_DEVICE_ALLOCATION_METHOD 20
MM_MAX_LOAD_TRIES 20
MODE 53, 61, 68, 69, 79
Model 113
Modify button 45
Modifying database objects 48
MONITOR 53, 70
Monitoring media activities 138
MOVE 52, 70

MOVE TAPE 129
Movement cycle 104, 122
Moving tapes 122
MType read by device window 101, 118
MType write by device window 102, 118

N

NAME 75, 80, 97, 98, 100, 101, 103, 105, 107, 112, 114, 116, 135
Name of associated scratch pool 104
Name of control host 113, 117
Name of device 99, 102, 115, 118, 119
Name of dropped media location 115
Name of host 97, 99, 119, 135
Name of jukebox 115, 117
Name of locality 97, 98, 99, 100, 101, 103, 105, 107, 108, 109, 113, 115, 117
Name of media location 115, 119, 132
Name of media pool 131
Name of media type 102, 118, 119, 131
Name of offsite media location 132
Name of onsite media location 132
Name of physical type 101, 131
Name of related saveset 136
Name of saveset 134
Name of tape 134
Name of user 108
Name of user field 109
Native accreditation 33, 35
Native host user accounts (proxies) selection window 107
Native user 21, 108
Navigating the GUI 36
NE 56
NEIGHBORHOOD 69
Neighborhood locality 34
NET_RECON_TIMEOUT 20, 142
NEWFILE 78
Non-cumulative incremental 89
NORMAL 131
NOTE DEVICE 52, 71
NOTE TAPE 52, 71
NOTEONLY 77, 81
Notification box 45
NOW 60
Number 105

O

Oasis RLM 24, 25
Object database 12, 15, 23, 93
Object name field 41
Object parameters 28, 29, 30
Object parameters on a UNIX Control Node 29
Object parameters on a Windows NT Control Node 28
Object parameters on an OpenVMS Control Node 30
Objects
- administration 12, 94
- database 35, 46
- device 18, 24, 46, 111, 115, 116
- device on host 19
- host 16, 23, 46, 94, 97, 98, 111
- jukebox 16, 24, 25, 26, 27, 29, 30, 46, 111, 112
- locality 15, 46, 94, 96, 106, 111, 141
- media 12, 93, 111
- media location 17, 24, 25, 26, 29, 30, 46, 111, 114
- media pool 17, 28, 29, 30, 47, 94, 95, 102, 103, 129
- media type 16, 27, 29, 30, 47, 94, 100, 101
- physical type 16, 47, 94, 99, 100
- privilege 16, 47, 106, 108, 109
- read media type 18
- related 47
- sample 15, 16, 17, 18, 19, 23, 93
- saveset 47, 129, 135
- segment 47, 129, 134
- standard 15, 16, 19, 93
- system policy 19, 47, 94, 104, 105
- tape 24, 25, 26, 47, 94, 111, 129, 130
- user 15, 47, 106
- user account 16, 24, 47, 106, 107, 108
- volatile 35
- write media type 18

OFFSITE 70
Offsite due date 122
Offsite duration 104
Offsite storage 86, 91
OK button 45
OMM Uninstall 31
OMM_REMOVE 32
ONSITE 70
Onsite due date 122
Onsite duration 104
OPEN SAVESET 52, 61, 72
OPEN SEGMENT 52, 72
Open windows box 34
Opened saveset 139
OpenMedia database *See* Object database
OpenMedia messages 143
OpenMedia model 12
OpenMedia server 141
OpenMedia service 20, 21, 22, 23, 31
OpenVMS Backup Node 31
OpenVMS Control Node 30
OpenVMS hosts 22, 32

Operation of the server 141
Operations menu 35
OPERATOR 16, 56, 113, 142
Operator controlled jukebox 23, 24
Operator privilege 109
Option 54
Organizing the OpenMedia database 93
OUT 56
Overflow count 104

P

PAGESIZE 75
Paper size 49
Parameter 54
Password 107
Password accreditation 36
Physical backup 90
Physical media types selection window 99
Physical name 113, 116
Physical type 16, 47, 94, 99, 100
  Comment 100
  Name 100
  Name of locality 100
Physical type window 100
POOL 70
Populating the database 15
Preferences 50
Preferred by device 114
Preferred by location 116
Print button 45
Print preview window 49
Printing reports 37, 49
PRIVATE 103
Private media pool 95
Privilege 16, 47, 106, 108, 109, 142
  Internal privilege 109
  Manager privilege 109
  Name of locality 109
  Name of user 109
  Operator privilege 109
  Privileges 109
Privilege window 109
Privileges selection window 108
Prompt detail window 129
Prompts 52, 67
PROTOCOL 20
PUBLIC 103
Public media pool 95
Pull down list 43

Q

Quantity of data 88
QUIET 61, 62, 63, 64, 65, 66, 68, 70, 71, 73, 77, 79, 80
Quiet mode 61
QUIT 53, 54, 68, 72

R

RANK 63, 115, 119
RAW 80
READ 77
Read devices 101
Read media types 116
RECONCILE JUKEBOX 73
Recovery 142
Recovery from lost users and privileges 109
Reducing windows to icons 39
Refresh button 45
REGISTER 53
REGISTER CHANGES 73
REGISTER PROMPTS 73
Related objects 47, 96, 98, 101, 114, 116
Related objects tab 37, 48
Relating multiple objects 48
Relating single objects 47
RELEASE 52, 73
Remove expired savesets 125
REMOVE SAVESET 52, 73, 125
Remove savesets selection window 125
Removing expired savesets 125
Removing OpenMedia 31
REPLY 53, 74
Reply to Prompt button 129
Reply to this Prompt button 129
Report 35, 49, 52, 58, 61, 74, 136
Reports menu 35
REQUEST 53, 76
RESERVE 52, 76
RESERVE DEVICE 76
RESERVE TAPE 76
Reserved device 138
Reserved tape 138
Resizing boxes 38
Resizing columns 41
Retention 86, 104
RETRY 62
Reviewing prompts 127
Robot manager 25, 26
robot_config 26
Robotic jukebox 23, 24, 25, 26, 27
Run Report button 45
Running site defined reports 140
Runs admed server 98

S

Sample object *See* Objects: sample
SAVE 75
Saveset 47, 129, 135
  Application 136
  Client date 136
  Client number 136
  Client text 136
  Creation date 135
  Expiration date 136
  Incomplete 135
  Max streams 136
  Name 135
  Name of host 135
  Name of related saveset 136
Saveset segments selection window 134
Saveset window 135
Savesets selection window 135
SCH:Robot 24, 26
SCRATCH 52, 77, 104, 123, 124, 131
Scratch expired tapes 124
Scratch media pool 95
Scratch tapes 137
Scratching expired tapes 124
Scratching tapes 123
SCRIPT 77, 80, 81, 82
Security 142
Segment 47, 129, 134
  Client absolute tape position 134
  Client relative tape position 134
  Index in stream 134
  Index on tape 134
  Name of saveset 134
  Name of tape 134
  Size kb 134
  Stream 134
Segment window 134
Selection criteria 56
Selection window 35, 36, 39
Selective backup 89
SEPARATOR 75
Server 12, 21, 141
SERVER DATABASE 53, 77
SERVER SHUTDOWN 53, 78, 141
Server startup 141
SERVER TRACE 53, 78
Service broker 12, 21, 22, 23
SERVICED 71
SET 52, 61, 78, 83
Setting checkboxes 41
Setting counter fields 41
Setting dates and times 42
Setting defaults 61
Setting object name fields 41
Setting preferences 50
Setting system policies 104
Setting up a HYPERtape Backup Node 31
Setting up a HYPERtape backup object 28
Setting up and managing the object database 93
Setting up OpenMedia 15
Setting up the OpenMedia database 23
SHOW 52, 61, 79, 83
Site defined report 35, 49, 74, 140
Size 134
Slider 42
Slot 132
Slot is wrong 132
SORT 63, 66, 67, 75, 77, 79, 80
Standard command buttons 44
Standard object *See* Objects: standard
Standard reports 74
Starting the CLI 53
Starting the GUI 33
Starting the OpenMedia service 21
Starting the OpenMedia service for the first time 21
Starting the OpenMedia service on OpenVMS hosts 22
Starting the OpenMedia service on UNIX hosts 22
Starting the OpenMedia service on Windows 95 hosts 22
Starting the OpenMedia service on Windows NT hosts 21
Static data 88
STATUS 52, 64, 74, 78, 81, 131
STATUS CONTEXT 81
STATUS DEVICE 81
Status menu 35
STATUS PROMPTS 53, 73, 81
STATUS TAPE 81
Stopping the CLI 53
Stopping the OpenMedia service 21
Storage management requirements 85
Storage management strategy 91
Storage media 89
Storing tapes offsite 86
STREAM 65, 72
Stream number 134
Syntax 54
System architecture 12
System policies selection window 105
System policy 19, 47, 94, 104, 105
  Char value 105
  Comment 105
  Name 105

Name of locality 105
Number 105
System policy window 105

T

Tab 37
Tagged selection criteria 58, 74
Tape 24, 25, 26, 47, 94, 111, 129, 130
Application 28, 29, 30, 133
Arrived date 132
Bad 131
Bar code 131
Cleaner 131
Client date 133
Client number 133
Client text 133
Comment 130
Comp expiry date 131
Due offsite date 132
Due onsite date 132
Error count 133
External label 130
First used date 133
Format 130
Internal label 130
Last loaded date 133
Last loaded for write date 133
Load count 133
Lock 131
Name of media location 132
Name of media pool 131
Name of media type 131
Name of offsite media location 132
Name of onsite media location 132
Name of physical type 131
Slot 132
Slot is wrong 132
Status 129, 131
Tape and saveset management capabilities 121
Tape contents 138
Tape location information 132
Tape statuses 139
Tape template 25, 26
Tape usage information 133
Tape window 130
TAPECONTEXT 65, 72
Tapes by external label 137
Tapes by external label selection window 130
Tapes by internal label 137
Tapes by location 137
Tapes by media pool 137
Tapes by usage 137
Tapes due offsite 137
Tapes due offsite selection window 126
Tapes due onsite 137
Tapes due onsite selection window 126
Tapes with errors 137
template_jukebox1-tape.template 25, 26
Time available for backups 88
Timestamp 58, 59, 60
Timezone 97
TITLE 76
Title bar 34
TO 74
TODAY 60
TOMORROW 60
TRACE_ERROR_HISTORY_KB 20
TRACE_MAX_FILE_KB 20
Tracking tapes 123
Type 103
Types of backup 89

U

Underflow count 104
Uninstall 31
Unique name 98
Unit number 117
Universally Unique Identifier *See* Uuid
UNIX Backup Node 27, 31
UNIX Control Node 29
UNIX hosts 22, 32
UNIX shell scripts 83
UNLIKE 56
UNLOAD 52, 81
Unload flag 29
UNTERMINATED 131
Updating the onsite media locations 122
UPGRADE 62
User 15, 47, 106
Comment 107
Mail address 107
Name 107
Name of locality 107
Password 107
User account 16, 24, 47, 106, 107, 108
Host 108
Name of locality 108
Name of user 108
Native user 108
User account window 108
User administration 106
User window 106
Users selection window 106
Using abbreviations 56
Using ASSIGN and DEASSIGN commands 82
Using browsers 39
Using media operations 124
Using media pools to organize tapes 95

Using pull down lists 43
Using SHOW and SET commands 83
Using the CLI 51
Using the DISPLAY command 55
Using the GUI 33
Using the Interrogations options 136
Using wildcards in selection windows 39
Uuid 59, 60

V

VERBOSE 61, 63, 64, 65, 66, 67, 68, 70, 71, 73, 77, 79, 80
Verbose mode 61
Version 35, 45, 53, 82
Viewing brokers for the locality 97
Viewing device preferences 115
Viewing device preferences by location 119
Viewing devices on the host 99
Viewing devices that read the media type 101
Viewing devices that write the media type 102
Viewing hosts that access the device 119
Viewing localities brokered by the host 99
Viewing media types read by the device 118
Viewing media types written by the device 118
Volatile object 35

W

Welcome window 23, 33, 45
Wildcard 39
Windows 95 hosts 22, 31
Windows NT 28
Windows NT (Alpha) hosts 31
Windows NT (Intel) hosts 31
Windows NT Backup Node 31
Windows NT batch files 82
Windows NT hosts 21
WITH 76
WRITE 77
Write devices 101
Write media types 116
Writing scripts 82

Y

YESTERDAY 60