CHAPTER 4

THE OS-9 UTILITIES

This section is an overview of the main utilities available with Professional OS-9, to help the user quickly identify which utility he will use for a particular job. It does not attempt to describe the operation or syntax of the individual utilities, nor are the lists of utilities exhaustive, as this information is available in the OS-9 User's Manual.

4.1 WHAT IS A UTILITY?

A utility is a program that performs some function to facilitate the use of the computer system, such as directory display, file copying, or disk formatting. By contrast, an application program performs some function that is an end in itself, such as word processing, data logging, or machine control. Utilities can be divided into three functional groups (although the distinction can be somewhat blurred):

a) Access to operating system functions from the terminal - for example, **load** to load a file of modules.

b) System management - for example, **tsmon** to allow the login of a user at a terminal.

c) General-purpose functions.

Because utilities are just programs, the user can write new utilities, or purchase additional utilities, either from Microware or from third-party suppliers.

4.2 UTILITY SYNTAX

OS-9 utilities have a common command line syntax that is simple to learn and use. The command line starts with the name of the utility. The remainder of the line consists of items separated by spaces or commas, and can contain only two item types (there may be any number of each type, in any order):

a) An option string, preceded by a '-' character.

b) A parameter, not preceded by a '-' character.

For example:

```
$ dir -ur fred
```

The '-ur' is a string of two options, 'u' and 'r'. The 'fred' is a parameter.

The utility scans the command line twice, first looking for options, and then looking for parameters. Therefore, in general options may be placed anywhere in the command line in any order. However, parameters may need to be in a defined order. For example:

```
$ copy fred henry -r
```

copies the file 'fred' to the file 'henry', overwriting any existing 'henry'. The **copy** utility determines which is the source file and which is the destination from the order of the parameters on the command line.

An option is used to change the default behaviour of the utility. The option is usually a single character, although some options are two characters. Usually the letter case of the option is not significant, but some utilities have so many options that some options have a different meaning in upper and lower case.

Options may themselves take a parameter. In this case the parameter directly follows the option character, with an optional '=' separating them. Multiple options characters may be grouped after one '-', but an option that takes a parameter must be at the end of the group. This applies even if the option parameter is optional. For example, the **pr** utility accepts the option '-z', to indicate that it should read file names from its standard input path. This option takes an optional parameter - the name of a file to read (instead of reading standard input). Therefore the '-z' option must always be the last in a string of options, even if the first form (no file name) is used.

```
$ pr -tz
$ pr -tz=file_list
```

In the example below, **pr** would assume that 't' is the name of the file containing the list of file names, not the '-t' option character:

```
$ pr -zt
```

Note that if a parameter or option string is to contain a space, comma, or one of the **shell** special characters:

( ) # ^ & ! < > * ?

it must be enclosed in single or double quote marks. The whole string must be enclosed. For example:

```
$ format -r /d0 -i=3 "-v=My Disk" -nv
```

The options that cannot take a parameter can be merged into one string:

```
$ format -i=3 /d0 "-nvrv=My Disk"
$ format -nvri=3 /d0 '-v=My Disk'
```

All well-behaved utilities will respond to the '-?' option, causing them to display helpful usage text rather than carrying out their normal function.

Utilities are written to make as few assumptions as possible about the objects they work on, to increase their usefulness. For example, most utilities can take their input from - and send their output to - any file, device, or pipe, by using the redirection capabilities of the **shell**. Pipes allow the functions of utilities to be combined. For example:

```
$ dir FRED -u ! grep -v "\.r$" ! pr -z >/p
```

The **dir** utility will list all files in directory 'FRED'. The **grep** utility filters that list, removing all file names ending in '.r'. The **pr** utility reads in the file names remaining, and prints each file to the device '/p'.

4.2.1 Formal Syntax Notation

Microware use a standard notation when describing the syntax for the options and parameters of a utility.

[ ] the enclosed item(s) is optional

{ } the enclosed item(s) is optional, and any number may be entered

< > the enclosed word or phrase is a description of the item to be entered

| separates possible choices

The item descriptions enclosed between '< >' characters are usually abbreviations. Example abbreviations (with examples of usage) are:

Abbr.	Meaning	Example
opt	option string	-z=file_list
path	pathlist	PROJECT/SOURCE/test.c
dir	directory name	PROJECT/SOURCE
name	symbolic name	apricot
str	text string	hello world
n	decimal number	1234

For example, the syntax for the **copy** utility might be described as:

```
copy {<opt>} <source path> {<source path>} | [<destination path>] {<opt>}
```

This indicates that zero or more options are permitted, placed anywhere on the command line, that there must be one source pathlist given, but any number more are permitted, or that optionally a destination pathlist may be given after the source pathlist. (If there is more than one source pathlist, a destination pathlist cannot be specified - the '-w' option must be used to specify a destination directory.)

4.3 UTILITIES FOR OPERATING SYSTEM FUNCTIONS

Many utilities exist to provide access to operating system functions. These are generally simple programs, converting the options and parameters given on the command line into appropriate operating system calls.

attr	Display or change file attributes.
copy	Copy files (but not directory structures).
date	Display the date and time.
deiniz	Detach an I/O subsystem (see the chapter on the OS-9 I/O System).
del	Delete file(s).
deldir	Delete complete directory structures.
dsave	Copy a complete directory structure.
dump	Display file contents in hexadecimal and text.

echo	Print a string to standard output (including binary data).
fixmod	Check or update module CRC and module header items.
free	Display disk free space.
ident	Display information about module(s) in a file or in memory.
iniz	Attach an I/O subsystem (see the chapter on the OS-9 I/O System).
link	Increment the link count of a module in memory.
list	Display a text file.
load	Load modules from a file into memory.
makdir	Make a new directory.
mdir	Display the module directory.
merge	Concatenate files to standard output.
mfree	Display the free memory map.
pd	Print the current execution or data directory pathlist.
pr	Output files with pagination (for printing).
printenv	Print the environment variables.
procs	Display existing processes.
rename	Change a file name.
save	Save modules from memory to a disk file.
shell	The command line interpreter.
setime	Set the date and time.
sleep	Sleep for a time (in ticks or seconds), or indefinitely.
tee	Copy standard input to standard output and multiple output paths.
tmode	Change options on standard input, standard output, or standard error path (terminal or printer).

touch	Set the "last modified data and time" of a file to "now".
unlink	Decrement the link count of a module.

4.4 SYSTEM MANAGEMENT UTILITIES

There are several utilities to assist with maintaining the system, including utilities to archive data, check operating system structures, and manage a multi-user system.

backup	Copy the whole of a disk.
dcheck	Check disk filing structure integrity.
devs	Display a list of currently initialized devices.
format	Format a disk (physical or just logical format).
frestore	Retrieve files from a tape (or disk) archive.
fsave	Archive to a tape (or disk).
irqs	Display a list of currently installed interrupt handlers.
login	Log in a user.
os9gen	Install a boot file on a disk (make the disk bootable).
tsmon	Monitor a terminal for user request to log in.
xmode	Examine or modify the options in a terminal or printer device descriptor in memory – affects all *subsequently* opened paths on the device.

4.5 GENERAL UTILITIES

Finally, there are many utilities that are simply useful general-purpose programs.

binex	Convert binary data to Motorola S-record format.
build	Copy input lines to a file.

cfp	Command line processor – repeat a command, substituting strings from standard input.
cmp	Binary comparison of two files.
code	Print the hex value of input characters.
compress	Compress a text file.
count	Count the lines in a file, or display a breakdown of the characters.
edt	Simple line editor.
exbin	Convert Motorola S–records to binary data.
expand	Decompress a text file (see **compress**).
grep	Filter input lines, passing lines that match (or do not match) a given pattern.
help	Display usage information for a utility (calls the utility with the '–?' option).
make	Compile/assemble/link multi–file programs.
qsort	Quick sort of lines in a text file by fields.
tr	Transliterate characters.
umacs	Screen editor – an implementation under OS–9 of the public domain **emacs** screen editor.