



KUBLER CONSULTING

6855 NE Arnold Avenue, Corvallis, OR 97330

Office Phone: 541-745-7457 fax: 810-885-1840 www.kublerconsulting.com

Command and Control: Unix Commands for MPE Users

By: Jeff Kubler
Kubler Consulting, Inc.
www.kublerconsulting.com

Introduction

You may have been an MPE user for many years and you probably hoped to continue being an MPE user for many more. I had my MPE system up and one day when my six-year came in and began randomly hitting keys on the console. Now there are a few combinations that might have caused some problems, but being a solid MPE system I had little to worry, so I let her continue. Had it been my UNIX system I would probably have jumped up and shooed her away from it. Some random combinations of keys can be lethal in UNIX. But the unfortunate truth is we will all have to get used to UNIX (Linux, WINDOWS, etc.) if we plan to stay in Information Technology (November 14th, 2001 announced the intended end of the HP 3000 5-years from the date). So as one who has gone before into the netherland of UNIX I give you my best information on using UNIX (principally the variant we know and love, HP-UX).

One thing that will make you feel comfortable as you move from MPE/iX land to UNIX HP-UX land is the appearance of the box. As HP developed the to platforms, the hardware side become more and more similar. I think their strategy was to eventually make MPE/iX a shell that ran under HP-UX. However, all that has changed. Now we must all prepare for HP-UX and beyond! This, of course, not withstanding our own knowledge that MPE/iX is superior to HP-UX in many ways.

This paper will discuss the following three areas:

1. The basic concepts needed to make the change from MPE/iX to UNIX and beyond.
2. The commands needed to accomplish tasks and what the commands relate to on MPE/iX.
3. The UNIX-based tools that will help in making the transition from MPE/iX to UNIX.

Basic Concepts

There are a number of basic concepts that must be understood for the new transfer from MPE/iX to HP-UX. There are however, some important informational pieces of data that will be helpful in talking UNIX. Although the boxes look very similar they are different at the core of their operating system. Some of the basic concepts are:

- The New Environment
- Operating System vs. the kernel
- Command Interpreter vs. shell
- MPE/iX Execution Queues vs. HP-UX Scheduler
- Job Management MPE/iX vs. HP-UX
- Manager.sys vs. root (Superuser)
- Hierarchical Structure vs. Tree Structure
- Scheduling capabilities of Stream vs. cron
- Special Characters in MPE/iX vs. HP-UX
- Logon UDC's vs. .profiles and Aliases
- MPE/iX File System vs. HP-UX file systems
- Layout of a HP-UX Application
- Help vs. Man Pages
- Hello vs. login
- File labels/codes vs. file extension/magic number

The New Environment

While the physical boxes that make up the new environment may look identical to the HP 3000, there are some basic differences that must be noted. Here are some of the key ones:

CASE MATTERS! - on MPE/iX, upper and lower case doesn't matter. However on HP-UX, it is critical that you remember to use the proper case.

HP-UX uses many cryptic commands, sometimes as short as two characters.

The CTRL + C will stop most commands.

The Ctrl + S will stop the display and the Ctrl + Q will restart that display.

When suspend character is set via 'stty susp ^Z' a process can be suspended with 'ctrl + z' command. Do the 'jobs' command to see the job that has been suspended. Then enter the 'bg %num' which is the number returned by the jobs command.

File names in MPE/iX are limited to 8 characters. This limitation does not exist in HP-UX. File names can use any character except the '/', '\$', '#', '@', '%', '(', ')', '=', and '&'. Can be up to 256 characters and contain alphanumeric or special characters like +, -, _, or ..

Disk space on MPE/iX is accounted for in sectors. A sector is 256 bytes. A byte is eight bytes and can hold a value of 0 to 255 (2 to the 8th power, less 1). HP-UX uses bytes, Mbytes, etc.

An example from listf, 2 shows sectors:

FILENAME	CODE	-----LOGICAL RECORD-----				----SPACE----			
		SIZE	TYP	EOF	LIMIT	R/B	SECTORS	#X	MX
STORESCH		72B	FA	112	112	7	32	1	1

An example from an `ls -al` shows the size in bytes:

```
-rwxr-xr-x  1 ccuser      olddb  1554496 Aug 27  2000 worklist
-rwxr-xr-x  1 ccuser      olddb  1799768 Aug 27  2000 worksheet
```

The Operating System vs. the kernel

HP-UX relies upon something called the “kernel” as the operating system. The kernel manages the memory, allocates system resources, maintains the file systems, and controls access to the computer. There are many parameters that affect the configuration and the size of the “kernel”. These are called kernel parameters and lead to an area of management called kernel tuning. There used to be parameters to tune within HP MPE/iX but several years ago the design of MPE got ride of much need to make changes to these. Since that time I have only heard of the need to make changes occasionally. The kernel of one system may be quite different from another. You can get a good description of the tunables within HP-UX at http://www.hp-partners.com/edaweb_public/html/technical_support/tuning.html. Some view the “tenability” of the kernel issue as a plus and others as a minus.

Command Interpreter vs. The UNIX Shell

Another area of understanding is that of the shell in UNIX. Unlike, MPE/iX there are a number of shells that can be used in HP-UX. The MPE/iX command interpreter is built in. The command interpreter or shell is just another program. A shell is a command language interpreter. The purpose of a shell is to translate command lines typed at a terminal into system actions. MPE/iX has just two “shells” the Command Interpreter and the POSIX shell. With HP-UX there are a number of “shells” that can be used: sh – bourne shell, ksh – korn shell, csh – C shell, etc. Some confusion can be caused from the multiplicity of shells because some commands may be specific to the shell. A number of commands are really programs unto themselves and will not be shell specific. In order to check for the likelihood of problems with shell specific commands you can check for a man page. These will exist for a program but will be documented under the shell if they are shell specific.

Here is a little more information on the various shells in HP-UX:

Sh – Bourne shell

The Bourne shell – This has a default prompt of "\$". It is the oldest of the shells and is found on every UNIX system. The Bourne shell is the oldest and exists on every UNIX

system. The Bourne shell is the default shell assigned to root, and is commonly used for writing command files or "scripts".

Developed by Steven Bourne of AT&T Labs.

Note: On HP-UX 10.X, sh runs the POSIX shell by default.

ksh - the korn shell

The Korn shell is a powerful interactive command language that is also noted as a high-level programming language. One of its strengths is its compatibility with the Bourne shell and its many features like the "C" shell. Features of the Korn shell include command history editing, line editing, filename completion, command aliasing, and job control.

This is the most MPE-like of the shells.

Developed by David G. Korn at AT&T Bell Laboratories.

csh - the C shell

The C shell The C shell has a default prompt of "%" (and sometimes "#"). It was developed at Berkeley and has lots of tricky features.

Developed by Bill Joy at the University of California.

T-C Shell

This is basically the c shell with a simple command-line editing.

Posix shell

POSIX or Portable Operating System Interface for Computer Environments is an international standard for operating systems. POSIX is a *standard* aimed at portability of applications, at the source level, across many systems. The POSIX shell on MPE/iX is the UNIX type shell available for use invoked via the 'sh'.

Switching Between Shells

The shell is set in the users .profile file and set at login. To try out another shell, simply type the shell's name at the command line and press **Enter**. For example, to switch to the Korn Shell, you could type **ksh** at the command prompt. By doing this you actually create a sub-shell (a shell running within another shell).

The Bourne shell is sh.

The Korn shell is ksh.

The C shell is csh. Located in /usr/bin/csh.

To permanently change shells, type chsh at the command line. You must enter your password and then the new shell name. For example, /bin/tcsh. The changes will take effect the next time you login. The form 'chsh username /pathtoshell' changes the default shell. Example:

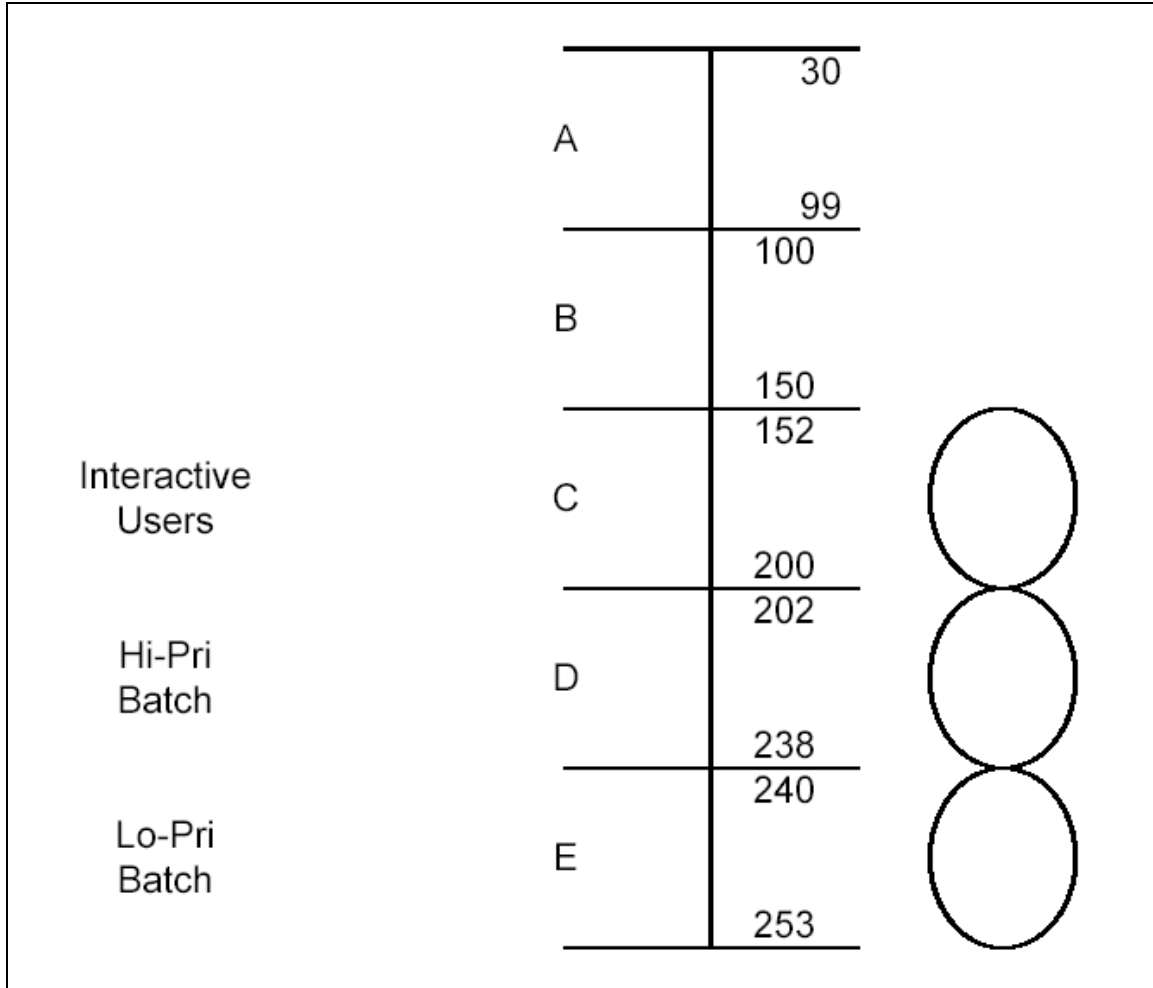
'chsh jeffk /usr/bin/csh'

MPE/iX Execution Queues vs. HP-UX Scheduler

Another area of interest for the new HP-UX user is the area of the method used to decide who gets the CPU and for how long. In MPE/iX we called this Queuing and the 5 default queues (and any others that you might create if you had the Work Load Manager) were called Execution Queues. Under HP-UX we have a different scheduling scheme that is called:

MPE/iX Scheduling

MPE/iX used the following scheduling plan:



MPE Process Priority Default Configuration

MPE users have benefited from the very handy distinction made between processes that were interactive and those that were batch.

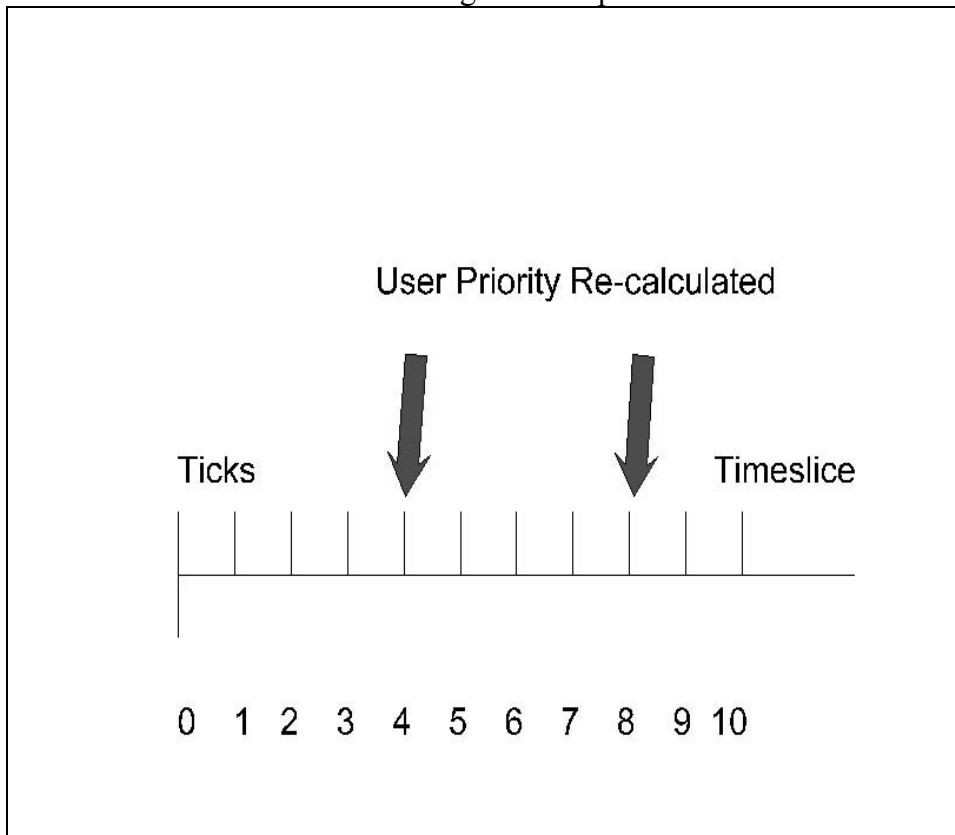
HP-UX Scheduling

HP-UX uses a different set of scheduling rules. Although HP-UX uses the same 0 to 255 numeric range for scheduling priorities there are basically two ranges that processes can run in, Real-Time and Time-Share. For Real-Time processes you have the following rules:

- Real-Time execute at highest priority.
- Real-Time processes are the most important system processes.
- Real-Time preempts any lower priority process
- Real-Time processes run until they sleep

Time-Share Processes have the following rules:

- Time-Share are time sliced 1/10 second. They use the following illustration to determine when Time slicing can take place:



HP-UX Process Priority Recalculation

- Time-Share are made up of system and user processes
- In Time-Share a higher priority process, that is ready for CPU, will pre-empt a lower priority process

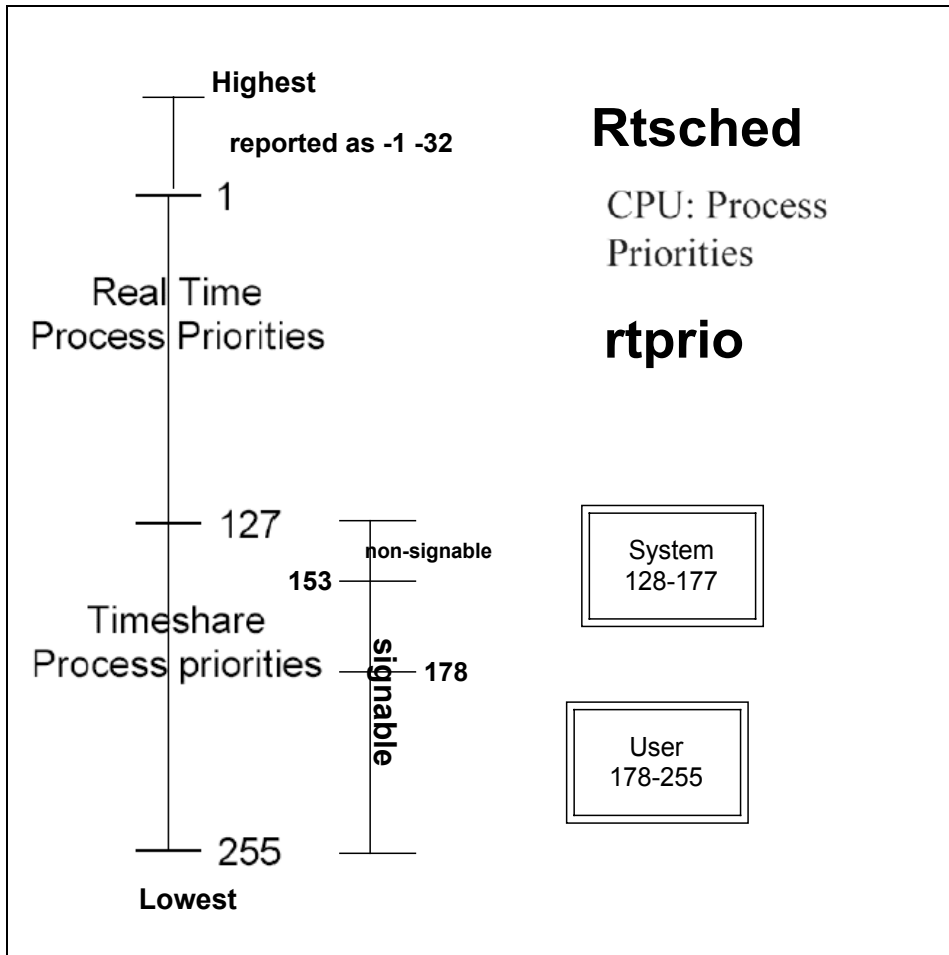
- In Time-Share, processes of equal priority will use Round Robin scheduling
- In Time-Share priorities are recalculated dynamically
- Nice values can be used to adjust priority calculations
- The statdeamon recalculates process priorities

Scheduling Synopsis:

Real Time are given a priority from 1 to 127. These are 1 wide and the highest priority process preempts all others. This highest priority will run until it sleeps or exits or is preempted by a higher or equal real-time process.

Equal process real-time priorities run in a round robin fashion until completed or this state of equality does not exist anymore.

The rtprio command can change a process to a real-time process.



HP-UX Process Priorities

The priority of a process is a dynamically recalculated (by the `statdaemon`) value assigned initially into different groupings based on whether the process is a highly important system process or a user process. The Scheduler gives a process, by default a 1/10 of a second. It divides this into ten ticks and calculates the user priority every 4 ticks according to this equation: $Newpri = ((\text{"recent CPU usage"} / \text{a constant}) + \text{basepri} + \text{nice value})$.

Nice - These are values (NOT NICE) that can be given to process to either increase the processes ability to compete for CPU or to decrease the processes ability to compete for CPU (the NICE value influences the calculation of the priority). A process with a 0 nice value has a priority of 178. A process with a 0 to 19 nice value will have its priority moved up (a 19 nice value would cause the process to begin with a priority of 159). And a process with a nice value of 39 has a priority of 197 ($178 + (39 - 20) = 197$). The NICE value is taken into account in the dynamic recalculation of priority.

Time-shared processes lose value in terms of priority as they execute. They regain priority as they wait their turn. This rate of decay is linear while the rate of regain for a processes priority is exponential. When the CPU load is high the exponential rate is decreased. When the CPU load is low this will be increased.

A child process is initially assigned the same priority as the parent.

When processes sleep, they are given a system priority. Therefore, a process set to sleep by the `sleep()` system call, is assigned a priority of 168, by PSPLEP (this explains an observation made while on-site of a *spectrum* process sitting at 168).

B3692A GlancePlus C.03.35.00 14:25:03 13000 9000/800 Current Avg High										
CPU Util	S							1%	1%	12%
Disk Util	FF							3%	5%	99%
Mem Util	S		SU UB			B		80%	80%	81%
Swap Util	U			UR R				58%	57%	58%
PROCESS LIST										
Process Name	PID	PPID	Pri	User Name	CPU Util (400% max)	Cum CPU	Disk IO Rate	Users=	Thd	Cnt
scopeux	2153	1	127	root	1.0/ 0.1	696.3	2.8/ 0.0	9.3mb		1
midaemon	2142	1	-16	root	0.4/ 0.3	3935.8	0.0/ 0.0	12.2mb		3
glance	7537	5914	154	jeffk	0.4/ 0.5	7.8	0.0/ 0.0	4.6mb		1
dtterm	7214	7213	154	ddemaree	0.2/ 0.2	4.8	0.0/ 0.0	1.1mb		1
dtterm	21986	21977	154	jared	0.2/ 0.0	39.9	0.0/ 0.0	3.8mb		1
vxfsd	36	0	138	root	0.0/ 0.2	3243.5	1.5/ 1.9	9.4mb		29
spectrum	7622	7588	168	saml	0.0/ 0.1	1.2	0.0/ 6.0	24.1mb		1
java	19897	1	168	dennis	0.0/ 0.1	908.2	0.0/ 0.0	22.6mb		16

SPECTRUM Process Priority

Some processes are said to be non-signable, that is they can't be interrupted during its operation. This can happen to processes during I/O operations when the process should not be interrupted until the I/O operation is completed.

In order to help ensure that the vitally important Spectrum processes can get the required CPU Options:

- Execute the Spectrum commands with a NOT NICE values (something between 1 and 19), or a “Nasty” value, causing its dynamic re-calculation.
- Execute process with the `rtprio` command. The `rtprio` command executes the process with a real-time priority.
- Execute the process with the `rtsched` command. The **Rtsched** executes *command* with POSIX or HP-UX real-time priority, or changes the real-time priority of currently executing process *pid*.

The usage of the NICE value to help or hinder processes in their demand for CPU is in some use. The use of `rtprio` or `rtsched` is rarely used due to possible performance implications to some users.

Of the `rt*` options (`rtprio` or `rtsched`), the `rtsched` command is the preferred command because it keeps within the overriding rule of maintaining POSIX compliancy.

Sample usage for options

Nice

```
'nice commandname'
```

Or

```
'nice -10 commandname'
```

Executes the program with a nice value of 30

Or

```
'nice --14 commandname'
```

Executes the program with a nice value of 6

Rtprio

```
'rtprio 127 commandname'
```

Rtsched

```
'rtsched -s SCHED_RTPRIO -p 127 commandname'
```

Note: The user running the `rtsched` command needs to have real-time scheduling privilege granted through `setprivgrp(1M)` [which I would set by editing `/etc/privgroup` because you need to preserve the global privilege of `CHOWN`].

Additional Options

HP markets a product called the Process Resource Manager (PRM) that may have some helpful aspects. The PRM provides a different mechanism for scheduling, based upon the FAIR SHARE SCHEDULER. PRM gives the CPU to groups of processes based on percentage allocations assigned by the system administrator.

Job Management MPE/iX vs. HP-UX

As you have noticed in the previous discussion, HP-UX lacks separate queues for interactive versus jobs. On HP-UX there is no job queue that places jobs in a lower queue and prefers the interactive user. Programs, commands and scripts executed from your screen take the screen unless forced into the background. They also run at the same area of process priority as all other processes. There are two things the user, who would like a script or command to act like a job. One step is to use the nice command described in the previous section to force the scheduler to treat the process with less preference for priority. The other is to use the 'nohup' command to put the command into the background and break the connection to the terminal.

Manager.Sys vs. Root (SUPERUSER)

The 'Superuser' in MPE/iX is the MANAGER.SYS logon. The System Manager could kill runaway processes, purge files, reset passwords, remove users from the system, etc. With HP-UX and UNIX the 'Superuser' is root. One problem with UNIX has always been that root circumvents all security checks and has free and total reign of the system. Root can delete any file, modify any programs, or change any user's password without an audit trail being left behind.

Because of the extreme power of root (and the fact that UNIX doesn't prompt you to confirm your actions) a good practice is for the system manager to only log on as root when absolutely required. One mistake is all it will take to prove this point (try accidentally `rm *` in an important directory to demonstrate). An alternative is to logon as a regular user, and then use the `su` command to switch to root when needed. The 'su' command allows a user to switch to another user.

Hierarchical Structure vs. Tree Structure

MPE/iX uses a very flat accounting structure called a Hierarchical Structure. HP-UX uses a Tree Structure. While MPE/iX has Accounts with many (at least one) group, HP-UX begins with the '/' directory and builds everything off this base directory in a tree structure (like DOS).

Special Characters in MPE/iX vs. HP-UX

In MPE/iX special character in use for commands consisted of the @ for a wildcard representing any character, a # for a numeric, a ? for one alphanumeric, the '<' for

redirection to an input, and '>' for a redirection to an output file. HP-UX has the following special characters for use in commands:

* - Matches any number of single characters e.g. `ls john*` will list all files that begin with john.

[...] - Matches any one of the character in the []

? - Matches any single character

`nohup` - Runs a process in the background leaving your terminal free

\$ - Values used for variables also `$n` - null argument

> - Redirects output

< - Redirects input to come from a file

>> - Redirects command to be added to the end of a file

| - Pipe output (eg: `who|wc-l` tells us how many users are online)

"..." - Turn of meaning of special characters excluding "\$,"."

`...` - Allows command output in to be used in a command line.

'...' - Turns off special meaning of all characters

Scheduling Capabilities of Stream vs. cron

The scheduling capability of HP-UX is one area that it may excel over what MPE/iX offers within the commands offered. MPE/iX has the ability to schedule in a rudimentary fashion using options on the STREAM command. With the STREAM command you can set a date, day of the week and time that you want something streamed. With HP-UX there is the cron command. This command allows for the scheduling of actions via a file that allows for the date, the time. Regularly scheduled commands can be specified according to instructions placed in crontab files. Users can submit their own crontab files with a crontab command (see [crontab\(1\)](#)). Users can submit commands that are to be executed only once with an 'at' or 'batch' command.

Several permutations of the cron facility are:

'crontab - e' to edit a schedule

'crontab - l' to list a schedule

'crontab - r' to remove a file from the schedule directory

Logon UDC's vs. .profiles and Aliases

What if you want to set up a specific environment or execute some set of commands upon logon? With the MPE/iX operating system you could set up a UDC (User Defined Command) that could consist of basic commands, other UDC's or command files, MPEX commands (an extension of MPE commands). This basically allowed the for the building of commands. The commands could be executed upon logon taking the user to a program or simply setting up the environment. With HP-UX a similar capability exists for setting up the environment using initialization files like the .profile file. Commands within this file are executed upon logon. Other files may exist that will also tailor the environment. There are two main files that are executed when you log on to UNIX, in this order:

.cshrc -- This file is read when C Shell starts up and when it is invoked from the initial login shell. For example, when you "escape" from Mail or run a shell script this file is source; .cshrc contains aliases and shell (non-environment) variables.

.login -- This file is read once, only when you first login; it contains commands and defines variables that only need to be executed at the beginning of the session: environment variables, terminal settings, commands such as new mail, etc. Sourcing this file will prompt the computer to call a new login screen.

Other .filename files can be sourced into use from within the .login or .profile files.

In order to create custom commands within HP-UX the user can create an alias. An Alias provides a means of shortening the name of a command or sequence of commands -- this is a feature you type an alias on the command line the C Shell replaces it with the command sequence to which it refers. The user can abbreviate or alter commands using an alias. An alias is usually setup in a .cshrc file so that they can be put into use at login. Form of the command is: 'alias newname command'

```
'alias dir ls -al'
```

or

```
'alias dir='ls -al'
```

will substitute the command 'ls -al' when you type dir at the prompt.

MPE/iX File System vs. HP-UX file systems

MPE/iX has one file system. HP-UX has several. The oldest is the HFS file system. The HFS (High Performance Filesystem) predates others and is fast but is not as fast to recover as the Journaled File System (JFS). JFS or the VxFS you will see under HP-UX is the safest file system. New tuning parameters make it just as fast as HFS and maintain the safety of the data.

Here are some of the major directories used under HP-UX:

Major HP-UX Directories

Path	Purpose
/	Root directory
/bin	Basic commands
/usr/bin	More commands and executables
/usr/etc	System administration commands
/usr/adm	Accounting files, system administration data files
/usr/spool	Spooling directories
/etc	Startup and configuration and some administration commands
/lib	C libraries

/dev	Device files directory
/tmp	World-writable storage for temporary files

Major HP-UX Directories

/ - the root directory, where everything starts.

/bin – Holds most executable programs. Other HP-UX commands are located in /usr/bin and other subdirectories.

/usr/bin – other commands

/usr – typically contains the applications.

/users – home directories for users. Likely the most dynamic subdirectory.

/dev – contains special files called device files. Device files make a connection between a piece of hardware and the file system.

/etc – contains most of the administrative commands.

/lib – Contains library file. These are related to programming languages and windowing environments.

/tmp – free-for-all place to be files temporarily.

Others:

/lost+found - This directory is where lost files end up. A system crash may cause a file to become disassociated with its directory. Fsck will put files here.

Laying Out an Application

Application layout in HP-UX is far different from application layout in HP MPE/ix. In HP MPE/iX an application might be laid out entirely in one Account and in one group under the account. Or it might be laid out with various Groups underneath on Account. In HP-UX there is a specific recommended manner for laying out the files that make up an application. Understanding this layout is important to maintaining and using any application. At one site I found the following layout that appeared to be a mirror image of what the group layout had been on the HP 3000:

```

drwxrwxr-x  2 saml      develop      8192 Apr 26 15:13 DATA
drwxrwxr-x  2 rich      develop      8192 Apr 24 14:39 JOB
drwxrwx---  2 karl      develop      8192 Apr 25 14:25 NOBACKUP
drwxrwxr-x  2 rich      develop          96 Mar 14 14:53 PGMS
drwxrwxr-x  2 jared     develop          96 Apr 26 16:20 REPORT

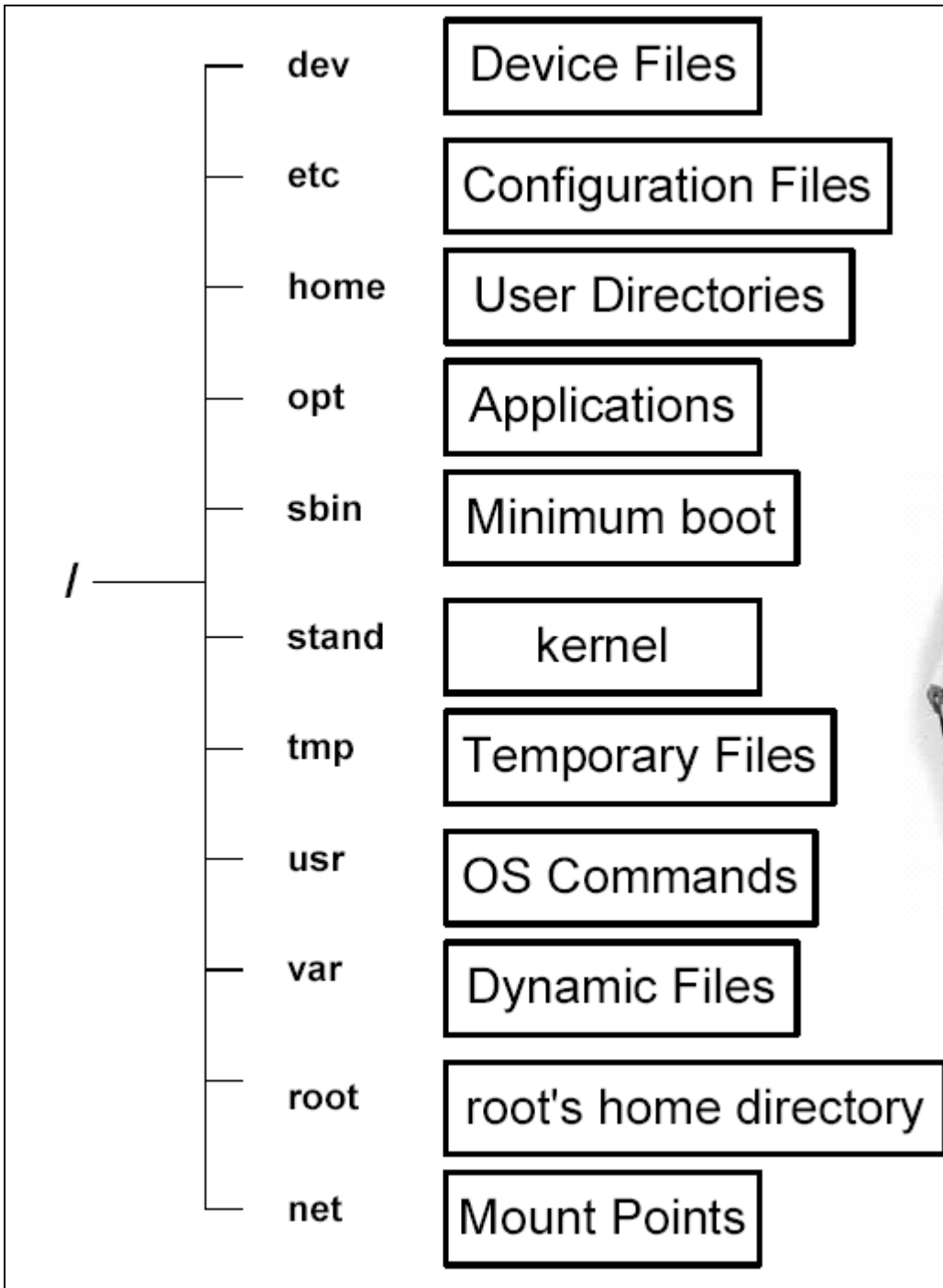
```

Subdirectory Names

In the Unix file system, /bin is for executables, /etc is for configuration files and scripts, /tmp is for temporary files, /lib is for library files and /var is for log files. If files appear in the root (/) directory they are for system files, if files appear in /opt/<app_name>/ they are for that application.

The system directory /usr is for system executables, /opt is for third party and optional vendor applications, /home is for user environment and login scripts. KCI consultants have found that it is a good practice to limit the amount of space in /home so that it cannot be used for used for project file storage. A separate file system for that project should be created and opened for access for that group of users. The Administrator should then insist that all project related files stay in that area. It is also a good practice to disallow a third party application from installing anything in the /usr file system.

The picture that follows gives a layout of the sub-directory structure in UNIX.



Sub-directory Layout

Help vs. Man Pages

MPE/iX users have become very familiar with the very user-friendly HELP command name syntax used. However, in HP-UX the man page becomes the substitute for the HELP command. The man command in conjunction with the command name will show

a page of help information. Man does have the ability to search commands for keywords which is very helpful:

`'man -k file'`

Here are some quick commands used in Man:

Q – quit

Enter key – go forward one line

Space Bar – go forward one page

B – go back one page

/ - search forward for a pattern

n – find the next occurrence of a pattern

Hello vs. login

The polite Hello user.account logon method for MPE/iX is replaced now with the simple logon name. The logon process is still one of the most expensive steps taken against the CPU resources. In MPE/iX the logon Hello would check the logon name, the passwords, check the session limit to see that it was not exceeded, display the welcome message, and run the option logon UDC's (to set variables, JCW's, file equations, and start applications). In HP-UX the logon does: a check to see that Daemon security is passed, checks the login name and password, checks for a trusted system, security configurations, checks disk quotas, executes shell initialization, and executes scripts (set variables, displays message of the day, displays copywrite, checks for mail status, checks for news status, sets umask and mesg, starts application or menu). MPE/iX uses NS/VT to connect while HP-UX uses telnet.

Practicing UNIX commands on MPE/iX

If you are on 5.0 and later, you have the POSIX shell (getting it setup correctly is an exercise in itself on 5.0 and 5.5 but supposed to run OK out of the box on 6.0 and above). If you do a

```
:setcatalog hppxudc.pub.sys;append  
or :setcatalog hppxudc.pub.sys;system;append (by the system manager)
```

you have the `:sh` command defined which invokes the shell. Otherwise you can `:sh.hpbin.sys -L` to invoke it. Inside the shell you are pretty much posix/unix-like and you can pipe. Traditional MPE commands, such as `:print`, have to be invoked via a wrapper:

```
posix/ix> callci 'print filename'|more
```

Because the shell doesn't know about the MPE print (MPE commands are by-and-large not simply executables in the current path; they are instead like 'internal' shell commands to the CI).

Basic Command Sets

The commands needed to accomplish tasks and what the commands relate to on MPE/iX
The basic command set in MPE/iX consists of HELLO, LISTF, SHOWJOB, SETVAR, SHOWME, STREAM, COPY, RENAME, CHGROUP, PURGE, PRINT, FILE, BUILD, and ABORTJOB.

The next level of commands is for more important users: ALTACCT, PURGEACCT, DELETESPOOLFILE, etc.

The basic command set HP-UX consists of a one word user name to logon, ls, ps, ll, cp, whoami, cd, mv, rm, echo, export, stty, PATH, and kill.

The next level of commands is for more powerful users: chmod, chown, su, rmdir, mkdir, etc.

Differing form for use in commands

A command can be just the straight command or the command plus some modifier. Both MPE/iX and HP-UX act this way regarding commands. In HP-UX there are optional parameters that can be added to most commands. When optional commands are used they can be prefaced by a '-' (hyphen). Therefore the command

```
'ls'
```

will list the file in the current working directory. The addition of the -s option will show the size option, and

```
'ls -s'
```

commands can also be strung together as

```
'ls -s; ps ; ll'
```

With this arrangement, each command is executed in sequence.

Piping

However, HP-UX supports something called 'piping' that MPE/iX, at least in the Command Interpreter format does not support. Piping or pipelines allows the user to string several commands together to make one hybrid command. Most commands in HP-UX will read from the previous input. Therefore a command can be constructed like

```
'ls -s | sort -n'
```

This command accomplishes an 'ls' and sorts it by a numeric sort rather than an alpha sort. The '|' is called a pipe mechanism. Any commands separated by the pipe mechanism are strung together in the shell. The standard output from each command is

run into the standard input of the next command. The beginning command gets its standard input from the terminal and the last command will put its standard output on the terminal.

Piping and Filtering Results

Because the ability to concatenate commands using the '|' to pipe them together some really good things can be done to reduce the output from your commands by filtering them together. Also, commands line the more command can be put together to help pause the output as in:

```
'ls | more'
```

or

```
'ps -ef | more'
```

The space bar is used to advance a page at a time and the enter key is used to move things ahead a line at a time.

Here is a summary of the things you can do in more:

- Scroll through the file a page at a time by pressing the space bar.
- Scroll backwards a page at a time by pressing **B**.
- Scroll through the file a line at a time by pressing **Enter**.
- Quit viewing the file and leave more by pressing **Q**.

One way of doing things would be to create a file, as an intermittent step and then look at the file output with a command as in:

```
'who > outwho.out'
```

```
'sort < outwho.out'
```

This would give the same results as:

```
'who | sort'
```

Some commands can be used to filter the output while in a pipe! The following commands can further filter the information:

- 'cut' – get specific columns or fields
- 'tr' – translate characters
- 'tee' – push output to files and stdout
- 'pr' – prints to stdout

Commands can be passed through the cut, tr, tee and pr command to filter and remove parts of the command that are not needed. An example of the cut command might be the following:

```
'ps -ef | cut -c49- | sort -d'
```

The above performs the `ps -ef` picking up the characters from 49 to the right and then sort the output.

```
'date | cut -c1-3'
```

Performs the `date` command and gets the letter from the day, as Mon, Tue, Wed, etc.

'`cut -cx-y`' – the 'c' option cuts columns or field. Here the x is the beginning character position and 'y' is the end.

The other form of `cut` looks like:

```
cut -f1 -d: /etc/passwd
```

This command will work on a file. It grabs the `/etc/passwd` file and considers the file broken into fields. The `-d:` sets the delimiter as a ":". By default the delimiter is a tab. Therefore the above command reads the `/etc/passwd` file and lists the user:

```
Root
```

```
Daemon....
```

```
cut -f1,6 -d: /etc/passwd
```

This version goes into the same file and gets the first and the sixth fields delimited by ":".

As in:

```
adm:/var/adm
```

```
uucp:/var/spool/uucppublic
```

The 'tr' command can convert or translate characters. In the format of '`tr -s [string1] [string2]`'. The 'tr' command can be a good predecessor to a `cut` command. One string can be replaced with another in the output by putting the string you want to change in the first position and the string you want to replace in the second.

```
$ who | tr "[:lower:]" "[:upper:]"
MARXMEIE   PTS/TB       FEB  2 13:04
KUBLER     PTS/TC       FEB 17 13:37
$
```

Changes the display from lower case to upper case.

The 'tee' command can be used to tap a pipeline in order to see the intermediary steps. Placing the `tee` command in front of another piped command followed by a file name will send the intermediary output to that file and go on to the next command.

```
$ who | tee unsorted | sort
kubler     pts/tc       Feb 17 13:37
marxmeie   pts/tb       Feb  2 13:04
```

Checking the file `unsorted` we find the unsorted output in the file.

File labels/codes vs. file extension/magic number

The File command takes a completely different form in HP-UX. In MPE/iX it performed the function of defining attributes for a file when built or on what device it was built or to redefine a file as being another file. The LISTF command with a ,2 would show the file type and any file codes.

With HP-UX the file command attempts to print a file after determining its file type. Without file labels or file codes (like MPE/iX) this is somewhat difficult. HP-UX looks at the file extension and at a magic number at the beginning of the file to check its type. There are no file labels or codes in HP-UX. A Magic Number is special data located at the beginning of a binary data file to indicate its type to a utility. Under Unix, the system and various applications programs (especially the linker) distinguish between types of executable files by looking for a magic number.

The file command also provided the capability to point the program about to be run to another file. On HP-UX this capability does not exist as it did in MPE/iX. However, the link command can point the file location in a specific directory to another location. "On UNIX the closest thing to the MPE file command for redirecting files is actually a symbolic link."

MPE File Types vs. HP-UX

MPE has a number of very defined File Types. These can be observed using the listf ,2 command or any listf variation above the 2. This information could be observed by looking at both the CODE columns and the TYP columns under the listf, 2. The figure LISTF, 2 Listing shows a sample output. The Codes could be: PRIV for database files, NMPRG for Native Mode Programs, PROG for Compatibility Mode Codes, SD for Self Describing files, etc. However, Codes and Typ's do not exist on HP-UX.

```
:listf ,2
ACCOUNT= KUBLER          GROUP= PUB

FILENAME  CODE  -----LOGICAL RECORD-----  ----SPACE----
          SIZE  TYP      EOF      LIMIT R/B  SECTORS #X MX
ABLDINTX  NMPRG  128W  FB      134      134   1    144  1  8
ALKEDCAT           80B  FA      416      416  16    144  1  1
ALKEDHLP           80B  FA     4551     4551  16   1456  1 27
AUTOCM     PROG  128W  FB       48       48   1     48  1  1
AUTOLED    NMPRG  128W  FB     2941     2941   1   2944  1  8
AUTOSPETH  NMPRG  128W  FB       428       428   1    432  1  8
CUSTREP1   RSPEC 4092B  VA        0         1   1        0  0  *
DBCOPY           80B  FA       71       71   3     32  1  1
DBEXPORT   NMPRG  128W  FB       622       622   1    624  3  8
DBINFO     NMPRG  128W  FB       513       513   1    528  3  8
DINVENT    SD      30B  FA        13        13 128    16  1  1
```

Listf, 2 Listing

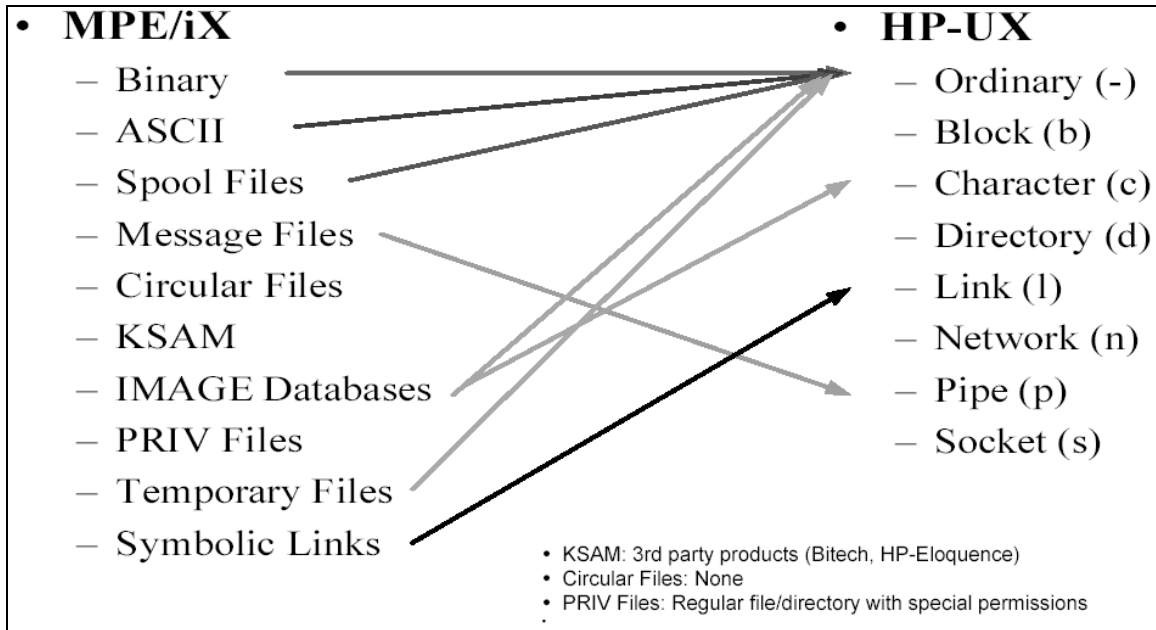
MPE/iX	HP-UX
– Binary	– Ordinary (-)
– ASCII	– Block (b)
– Spool Files	– Character (c)
– Message Files	– Directory (d)
– Circular Files	– Link (l)
– KSAM	– Network (n)
– IMAGE Databases	– Pipe (p)
– PRIV Files	– Socket (s)
– Temporary Files	
– Symbolic Links	

MPE/iX to HP-UX File Type Comparison

Essentially, everything on HP-UX is a file!! Therefore, file types don't exist. A – precedes normal ordinary files. A (b) refers to a file block file used to communicate with block devices like disk drives. The (d) refers to a directory file. A cd can be done to move to the directory file. An (l) refers to a link of some kind. Links usually take place when a file needs to be moved but some program expects to find a file there. This may happen when a file needs to be moved because of limited space in the file system. The file can be moved and the link directs the application to find the file. The (n) is a networked file.

Here is an example of a block file:

```
$ ll /dev/dsk/*  
brw-r----- 1 bin    sys    31 0x002000 Apr 18 2002 /dev/dsk/c0t2d0  
brw-r----- 1 bin    sys    31 0x01f000 Apr  4 2002 /dev/dsk/c1t15d0  
brw-r----- 1 bin    sys    31 0x022000 Apr  4 2002 /dev/dsk/c2t2d0
```



File Types to HP-UX Types

The “File Types to HP-UX Types shows the equivalences to MPE/iX files. Some files have no equivalences. Notice the KSAM files don’t have a direct equivalence. Neither does PRIV files that make up IMAGE datasets and circular files.

More on Files

Commands will often need the name of a file as an argument. Pathnames in HP-UX consist of several components separated by a ‘/’. Each piece is the name of a pathway until the filename is provided as in

```
/etc/motd
```

This finds the file motd in the /etc subdirectory. /etc is a subdirectory of the root directory ‘/’. When the ‘/’ is used in this manner we are using the ‘absolute’ pathname. If the ‘/’ is not used in the beginning the search starts from the current directory. By default, this beginning point is the home directory. File names without a directory mean that the file is located in the current directory.

File names consist of a number of alphanumeric characters and/or numeric characters. They can have file extensions past the ‘.’ And multiple ‘.’s may be used. All characters can be used in a name save that of the ‘/’. Common file extensions are:

.profile, .ksh, .* -- Any file with a .filename is likely to be an initialization file. Initialization files are executed upon logon and used to tailor the environment to the users needs.

HP-UX sets up a `.cshrc`, `.exrc`, `.login`, `.profile`, and `.sh-history` files upon the creation of a user, in that users home directory.

A number of different extensions are common and are helpful to know when getting around in HP-UX. Here are some of them:

- `.c` for the source of a C program
- `.o` for the object file
- `.errs` for the errors from the file
- `.output` for the output of the run of the program
- `.h` for header files
- `.gzip` for zipped files
- `.gz` is the Gnu version of zip. It is a compression method developed for use on UNIX systems.
- `.tar` for tar files

The `*` can be used in place of the `@` for alphanumeric characters that was used in MPE/iX. The `*` is a wildcard for any number of alphanumeric characters. The character `?` matches any single character in a filename. Thus the `ll ?` will yield any single character filename. An echo command in the following format

```
'echo ? ?? ???'
```

Will show all single character filenames, all two-character filename and all three-character filenames.

Two other characters worth noting are the `[` and the `]`. These allow for a list of characters placed in between them. In the following `ll` any files with `j` in the first position will be selected.

```
'll [j]'
```

Finding files

Are you looking for a file? If you know where it is located you can simply `cd` to the directory. If not you can use the `find` command. The `find` command has the following syntax

Find pathname filename argument (this usually `-print`)
Eq.

```
'find / -name filename -print'
```

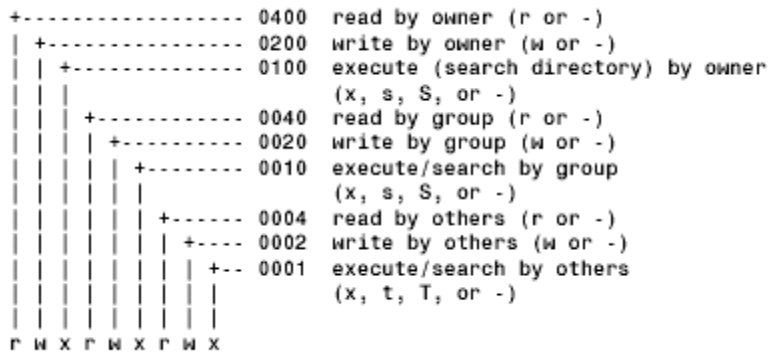
File Ownership

File ownership is setup by the user ID of the logon who creates the file. The owner can determine who may read, write or change the file, or execute the file. However, the ownership can be changed via the 'chown' command.

The 'ls -al' command will reveal who can read, write or execute a file. The following shows a files ownership and list of access capabilities

```
-rwxrwxrwx 1 root sys 506 May 1 18:48 testfile2
```

The file will allow anyone to 'rwx' or read, write and execute. The pattern is repeated three times, on for the owner, the group, and the rest of the system wide users(all others). The 'r' means that owner, or group or system can read a file, the 'w' can write to the file and the 'x' deals with executing from the respective group. The following figure shows the usage of the 'r', 'w', and 'x' (these are the most common but there are other letters that can be in these positions along with the '-').



The first letter indicates the entry type with the following meanings: d – directory, b – block special file, c – character special file, l – symbolic link, p – Fifo special file that can also be called a “named pipe”, n – network special file, s – socket, - - ordinary file.

In file listings (such as the listings shown by the **li** or **ls** command), the three groups of users are always represented in the following order: user, group, and others. If you need to find out your group name, the **groups** command shows all the groups for a user ID.

```
-root-> id
uid=0(root) gid=3(sys)
```

Every user ID is assigned to a group with a unique group ID. The system manager creates the groups of users when setting up the system. When a file is newly created, the operating system assigns permissions to the user ID that created it, to the group ID containing the file owner, and to a group called others, consisting of all other users. The 'id' command shows your user ID (UID), group ID (GID), and the names of all groups your user belongs to.

What can a file do?

Files in HP-UX don't have a code as in MPE/iX. Nor do they have a Type or a record count. Files have a size and sometimes an extension. However, there is a command that can be used to help tell what the file can do. The file command will help tell what a file can do:

File filename – will determine the file type.

Examples of the information that can come from the command are:

```
JEFFD1:      ascii text
OUTPUT:      data
STORESCH:    pascal program text
ecoserve.tar: tar file
gatherer.sh:  commands text
gnuplot:     directory
jeff44:      awk program text
jrkt:       English text
mallocmegs.c: c program text
nohup.out:   nettl binary Log file -version 20039
unxlink2:    PA-RISC2.0 shared executable dynamically linked -not
stripped
```

Output Redirection

Both MPE/iX and HP-UX allow the user to send output from a command to a file. In MPE/iX this is accomplished by:

In HP-UX this is accomplished by:

```
'll > filename'
```

The output from the ll command is written to a file. Additionally, input can be sent to a command using:

```
'cmd < file'
```

The above reads from the stdin file. Appending to an existing file can be done via:

```
'cmd >> file'
```

The output or stdout from one command can be sent to another in the following form:

```
'cmd | cmd1'
```

The error output (stderr) can be redirected using the following:

'cmd 2> file'

MPE/iX Command vs HP-UX

The following table provides a translation table for the MPE/iX commands you are familiar with to the HP-UX commands you will need.

MPE/iX	HP-UX
•ABORTJOB	•kill, kill -9
•ALTACCT	•chmod, passwd
•ALTGROUP	•chmod
•ALTSEC	•chmod
•BYE	•exit
•CHGROUP	•cd
•CI	•sh, csh, ksh
•COPY	•cp
•EDITOR, QUAD, QEDIT	•vi
•FCOPY	•cp, dd, lp
•FILE	•(implied), ln
•HELLO user.acct	•login user
•HELP	•man
•HPSEARCH, MPEX LISTF	•grep
•LISTF	•ls, find

•LISTGROUP	•ls -l
•LISTREDO	•history, cat sh.history
•NEWACCT	•mkdir
•NEWGROUP	•mkdir
•PRINT	•more , cat, lp
•PURGE	•rm
•REDO	•esc k,
•RELEASE	•chmod 777
•RENAME	•mv
•REPORT	•bdf, du, df
•RESTORE	•tar, frecover, cpio
•RUN	•(implied)
•SETJCW	•=
•SETVAR	•=, csh – setenv, sh, ksh NAME=value; export NAME
•SHOWGROUP	•pwd
•SHOWJCW	•echo
•SHOWJOB	•ps -ef, who, users
•SHOWME	•whoamI
•SHOWOUT	•lpstat -t
•SHOWTIME	•date

•SHOWVAR	•echo, printenv, env, %echo \$NAME
•SHUTDOWN	•shutdown -0 -y, reboot -h
•STORE	•tar, fbackup, cpio
•STREAM	•at, bg, crontab, nohup
•SYSDUMP	•make_recovery
•TELL	•talk, wall, write
•WARN	•talk, wall, write
•WELCOME	•/etc/motd
•Superuser=MANAGER.SYS	•Superuser=root
	•ioscan
	•lvcreate
	•lvextend
	•lvremove
	•mount
	•pvcreate
	•sam
	•swapinfo
	•vgcreate
	•vgextend
	•vgremove

Purgedir	•Rmdir
Store	Cpio, tar, fbackup
	Which
Dscopy	ftp
God(mpex)	Su
Deletespoolfile	Cancel, lprm
Build	Touch
UDC file option logon	sh .profile ksh .profile and file in ENV variable csh .cshrc, .login, and logout

Figure 1 - Command Matrix

For the sake of discussion the discussion of commands is divided into a number of levels. Level 1 is a basic set of commands organized into their functions.

HP-UX Commands - Level One

There are several helpful things to remember regarding HP-UX commands. The first is that commands are case sensitive. Unlike MPE/iX an 'n' and an 'N' are not the same. When commands are shown they should be entered exactly as they appear.

Command Descriptions

Logon – unlike MPE/iX no polite :HELLO is required by UNIX. Simply enter your user name (this must be provided to you) followed by your password.

What happens at login?

After establishing a connect HP-UX provides a prompt:

```
login:
login: kubler
Password:
Last successful login for kubler: Thu Feb 07 07:54:18 PST8PDT 2005 on pts/tc
```

Last unsuccessful login for kubler: Wed Feb 6 10:16:19 PST8PDT 2005 on pts/td
Please wait...checking for disk quotas

(c)Copyright 1983-2000 Hewlett-Packard Co., All Rights Reserved.
(c)Copyright 1979, 1980, 1983, 1985-1993 The Regents of the Univ. of California
(c)Copyright 1980, 1984, 1986 Novell, Inc.
(c)Copyright 1986-1992 Sun Microsystems, Inc.
(c)Copyright 1985, 1986, 1988 Massachusetts Institute of Technology
(c)Copyright 1989-1993 The Open Software Foundation, Inc.
(c)Copyright 1986 Digital Equipment Corp.
(c)Copyright 1990 Motorola, Inc.
(c)Copyright 1990, 1991, 1992 Cornell University
(c)Copyright 1989-1991 The University of Maryland
(c)Copyright 1988 Carnegie Mellon University
(c)Copyright 1991-2000 Mentat Inc.
(c)Copyright 1996 Morning Star Technologies, Inc.
(c)Copyright 1996 Progressive Systems, Inc.
(c)Copyright 1991-2000 Isogon Corporation, All Rights Reserved.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in sub-paragraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
3000 Hanover Street
Palo Alto, CA 94304 U.S.A.

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

You have mail.

\$

Logon Example

At login, the system initiates the . files for .profile. This depends upon the shell that is used for the default shell.

Getting help

The HELP command, with its variations for showing EXAMPLES and 'man' - like the help page is very useful. Options exist in the man page to search for a key word and print specific portions of the man page. Man is used as the command with the command you want to understand listed following man.

'man -k keyword'

will show those commands that have a keyword specified. Generally man is used simply as:

'man commandname'

Moving from directory to directory

Moving from one account and group to another could be accomplished in MPE/iX via a new login or via the CHGROUP command. On HP-UX and UNIX the 'cd' command allows for the moving from one directory to another. The directories are always separate be '/.

New users to HP-UX are assigned a login ID and a directory that they will own and control. These are usually assigned under the /home directory. The directory will have the same name as the users login. Subdirectories can be created under the users directory. Initial logins get placed in their home directories.

'cd' - change the current working directory. Used in the form cd intended_directory. Cd \$HOME returns the user to their home directory. 'cd directory name' moves

Absolute versus Relative Reference

When you do a 'ls -al' the first two lines you see are:

```
$ ls -al  
total 7694  
drwxr-xr-x  8 kubler  users      2048 Feb 16 09:39 .  
drwxr-xr-x 98 root    root       2048 Jan 25 09:36 ..
```

The . and the .. entries. These entries are created automatically and serve to allow the use of relative path names. While the . represents the current directory position, the .. represents the directory above the current directory. This is sometimes called the parent directory. Examples of absolute location are /home, /, /home/jeffk. Examples of relative directory location are .., ../ (to / directory), ../../ (to / directory), and ../tmp (to the /tmp directory).

Listing files

It is always important to be able to list files that are being worked with. In MPE/iX the Listf command (and the many additions that could be added to it via the MPEX program) are vital to commands with many helpful variations to show file information. On HP-UX and UNIX the 'ls' and 'll' commands fill this need.

'ls' - listing files contents of directories. Several permutations are lsf, lsr and lsx are used to invoke the ls command with the -f, -r and -x options.

'll' - List the contents of a directory. Common form is 'll -al' to show contents plus access options, owner/creator, size and creation date.

Checking processes

MPE/iX didn't have more than the SHOWJOB to view who was logged on. You had to purchase Glance or SOS to really see what was happening to processes and users. HP-UX and UNIX in general have a number of very helpful commands and utilities that make find out what is happening to a process or user accessible without a purchase of a tool. The 'ps' command is one command that shows what is happening on the system.

'ps', listing processes, Process status report. Common form is 'ps -ef' to show all processes. The 'top' command, however is not similar to anything that MPE/iX users have used without a purchase.

'who' – shows who is on the system.

```
-root-> who
root      ttyt1      Jun 12 11:06
jeffk     ttyt2      Jun 12 05:35
Fred- /root/home/root
```

'top' - displays and updates information about the top processes on the system.

```
System: Fred                               Wed Jun 12 11:07:14 2002
Load averages: 1.07, 1.12, 1.12
86 processes: 85 sleeping, 1 running
Cpu states:
  LOAD   USER   NICE    SYS   IDLE  BLOCK  SWAIT   INTR   SSYS
  1.07   9.9%   0.0%   4.0%  86.1%  0.0%   0.0%   0.0%   0.0%

Memory: 5600K (3656K) real, 31284K (13424K) virtual, 1716K free Page# 1/7

  TTY   PID USERNAME PRI NI   SIZE   RES STATE   TIME %WCPU %CPU COMMAND
root  1197 root      154 20   500K   592K sleep 29:42  8.85  8.84 dtgreet
root  1185 daemon  154 20  2504K  812K sleep 17:51  5.39  5.38 X
pty/ttyt1 1424 root      179 20   268K    0K run    0:00  7.01  0.34 top
pty/ttyt1 1402 root      158 20   520K  148K sleep 0:00  0.29  0.27 sh
root     3 root      128 20    0K    0K sleep 0:30  0.12  0.12 statdaemon
```

Checking the environment

Sometimes you just need to check the environment. On MPE/iX you might do a SHOWME to see how you are logged on, or check the file equations with the LISTEQ command. While the ability set file equations does not exist in the same manner on HP-UX does have dynamic linkage available via the ln command. The echo \$PATH shows the path set for the user. Programs to be utilized by the user should normally be in the path (The 'whoami' command shows how the user is logged on, the uname command does a similar function to the SHOWVAR HPSUSAN command in MPE/iX).

'whoami' - prints the effective current user id

'date' - display or set the date and time.

'uname' – This command displays information about computer system. Common form is 'uname -a'.

'echo \$PATH' – this shows the path that has been set for the user.

'pwd' – to check the current directory.

```
-root-> echo $PATH
/sbin:/usr/bin:/usr/bin/X11:/etc:/usr/contrib/bin:/usr/local/bin:/usr/lib:/opt/robelle/bin:/root/home/root:/root/home/root/bin:.
```

'which' - will locate a program file (including aliases and paths). The 'which' command checks down the \$PATH to find which program is being run.

Copying, Moving and Deleting files and directories

Just as in MPE/iX, files in HP-UX can be copied (instead of the COPY command use cp), renamed (instead of the RENAME command use mv to move the file), removed (instead of the PURGE command use the rm command). As in MPE/iX, accounts and groups can be created and removed. However, instead of NEWACCT and NEWGROUP use mkdir and instead of PURGEACCT and PURGEGROUP use rmdir.

'cp' - copies files and directories.

'mv' - move or rename files and directories.

'rm' - remove files or directories. This command is very dangerous. The root user could wipe out everything with rm *.*. {no inquiry is made to double check your intentions}

'mkdir', 'rmdir' - create and remove directories.

Making life easier

One great feature of MPE/iX has always been the ability to see what had been done and then redo the command easily with either the DO command or the REDO command.

With most shells the 'history' command will show the previous commands executed.

The combination of esc k will repeat the previous command.

'history' - in the c shell, the history command shows a list of previously executed commands. This is like the 'listredo' command in MPE/iX. In the **cs**h, **ks**h use history, in the bourne shell use **sh #fc -1**.

esc k - to repeat previous command. Repeating or redoing commands differs depending upon the shell you are utilizing.

Redoing commands - The redo command varies slightly from shell to shell. In the **ks**h use the \$r, in the c shell use the **cs**h %!, **sh** none.

ksh ESC, then vi commands such as k (up), j (down), i (ins), 0 (home), \$ (end), x (delchar), a (append), then Return.

Don't use the arrow keys to move where you want but rather utilize the space bar and the backspace keys.

Redoing the c shell summary:

csh !\$ anywhere means *last thing from last command*

csh !:5* means *argument 5 thru last of last cmd*

csh !* is short for !:1*, *all the arguments of last cmd*

csh ^li^il *replaces li with il in last command*

'sort' - Allow for the sorting of files, etc. This is often used in conjunction with other commands.

Jobs can also be schedule in batch to begin at a latter time.

at, batch - execute batched commands immediately or at a later time.

Example:

```
$ at -f $HOME/testscript -t200502171100.00
job 1108666800.a at Thu Feb 17 11:00:00 2005
$
```

Another helpful capability is the backwards single quote. Used in a command as in `command` the command gets executed and the phrase within the ` quotes will be replaced with the contents of the command. As an example:

```
$ ls -al `ls test*`
-rw----- 1 kubler users 0 Jan 20 2004 testfile1
-rw----- 1 kubler users 0 Feb 16 14:33 testfile3
-rw----- 1 kubler users 10381 Feb 16 12:08 testls
-rwxrwxrwx 1 kubler users 171 May 3 2002 testscript
$
```

This is somewhat similar to indirect files in MPE/iX.

Printing files to the screen

The PRINT command is not available, as it is in MPE/iX. However, on HP-UX and UNIX there are a number of commands to print files.

‘cat’ - will display files or join them together.

‘head’ – this shows the first few lines.

‘tail’ - this shows the last few lines.

‘echo’ - echo (print) arguments.

The ‘cat’ command will display files. It will also put them together as in ‘cat file1 file2 > file3’. This command takes existing files file1 and file2 and puts them together in file3.

Looking for things

Finding the location of needed files could be done in MPE/iX via the LISTF command.

In HP-UX one way to find files is the ‘find’ command. Strings could be searched for in MPE/iX via an extension of MPEX. That command was ‘PRINT

@.@.@;search=cl{caseless}”string” {the string that is being looked for}.

‘find’ - finds files. Common form is ‘find /{this brings you to the root directory} –name {looks for the file name} filename –print {to print the results to the terminal}. An interesting note about ‘find’ is that it will drop into the system mode for some of its processing.

‘grep’ – the grep command is often used in conjunction with other commands. One example is to look within a set of files.

Fixing the Environment

Sometimes things don’t work. In MPE/iX the SHOWVAR command, LISTEQ, could be used to help debug the issue. In HP-UX the ‘printenv’ command shows the variables that are set in the environment. The ‘export’ is used in some shells to put the variable into use once set.

‘printenv’ – this command prints out the environment variables.

‘setenv’ – In the c shell this command sets a variable. Used in the form ‘setenv jeff 1’.

'export' – Used in conjunction with the form 'variable name = xx' the 'export' puts this into effect.

'chmod' - change file mode access permissions. 'chmod 777 filename' gives everything to everyone. Be careful of this! 'chmod' can be used to effect the addition or subtraction of capabilities in the form,

'chown' – change the files owner. The form of 'chown' is generally 'chown filename newowner'.

'kill' - terminate a process. 'kill -9' terminates everything about a process.

Sometimes simply adding the correct subdirectory to the path may fix the problem as in PATH=\$PATH:/jeff/new. This keeps the current path and adds /jeff/new to the path.

Setting up the terminal

Setting up the terminal environment is a key component to getting things working properly. Also, setting the correct PATH is a critical piece of getting an application to run. In MPE/iX there wasn't much that needed to be done except ensure that the terminal emulator was set up properly. In HP-UX the .profile file probably contains a line setting up the terminal for proper erasing of characters, back space, etc. A line such as 'stty erase "^H" kill "^U" intr "^C" eof "^D"' will set up the terminal keys.

'stty' - set the options for a terminal port

PATH – this sets the path for the programs and files that may be accessed easily along the path. Use 'echo \$PATH' to see what is set for the path user is set to.

Commands, programs, utilities to help with the Transition

The UNIX-based tools that will help in making the transition from MPE/iX to UNIX Configuration Tools

Sam – System Administration Manager. The SAM application will do almost any maintenance and configuration function required on HP-UX.

HP Eloquence has a similar structure to MPE/iX's TurboImage. SUPRTOOL from Robelle also works on HP Eloquence.

Aliases can also be created to make commands similar to those on MPE/iX, however, this may actually prove to be a hindrance. It may be acceptable for some users.

Editors on HP-UX

VI – is the Universal Editor on UNIX and on HP-UX. VI is not as friendly as other editors you might find but it is universal in its availability on UNIX. Just as EDITOR is universally available on MPE/iX and therefore a wise thing to learn, VI is also a wise thing to know. Once you do you may want to get something a bit easier. Robelle's fine editor QEDIT exists on HP-UX (as does SUPRTOOL for ORACLE and ELOQUENCE databases).

In order to run "VI" simply type 'VI' and the file name you wish to work on:

'vi jeffsfile'

Some Simple VI Commands

Here is a simple set of commands to get a beginning VI user started. There are many other convenient commands, which will be discussed in later sections.

a - enter *insert* mode, the characters typed in will be inserted after the current cursor position. If you specify a count, all the text that had been inserted will be repeated that many times.

h - move the cursor to the left one character position.

i - enter *insert* mode, the characters typed in will be inserted before the current cursor position. If you specify a count, all the text that had been inserted will be repeated that many times.

j - move the cursor down one line.

k - move the cursor up one line.

l - move the cursor to the right one character position.

r - replace one character under the cursor. Specify *count* to replace a number of characters

u - undo the last change to the file. Typing u again will re-do the change.

x - delete character under the cursor. *Count* specifies how many characters to delete. The characters will be deleted after the cursor.

Scripts

One of the most powerful aspects of HP-UX and UNIX is the ability to create Scripts to perform complex or repeated tasks. This is also a capability within MPE/iX. The following is a sample Script. This is a typical Unix shell script skeleton:

```
#!/bin/sh
# First line in the script, the script now runs with Bourne shell
# Name: demo.sh
# File name for the script. I use *.sh for Bourne shell scripts,
# *.csh for C-shell scripts, etc.
# Purpose of the script, just few lines, with references to
# documents
# Created: JRK, 17-JUL-2001 at 16:10
# Who wrote this and when, also short update history and exact
# time when the script was last changed.
cd $HOME/RunMyScript
# Directory where the script is to be run. $HOME is a shell
# variable containing a part of the full path, you can use
# full absolute path instead of variable, like /user/local/MyScript
echo `date` " Script _demo.sh_ started " > sam_script.summary
echo `date` " Script _demo.sh_ started " > sam_sh.log
echo `date` " Script _demo.sh_ started " >> sam_cumulative_sh.log
# Now this script created or overwrote two files, sam_script.summary
```

```

# and sam_sh.log and added one line to file sam_cumulative_sh.log
# The purpose of *.summary is to collect messages for the
# script users.
# The purpose of *.log is to collect technical information for this
# run in order to help fix possible problems.
# The last file contains the whole history of runs with this script.
# Remember to empty (delete) this cumulative log now and then.
# --- Do the real work here ---
# --- Send results and run reports ---
# FTP
# mail
# --- End ---
# End of demo.sh script
# It is quite useful to indicate the real end of the script !

```

Scripts rely heavily on environment variables. The most commonly used variables are TERM, DISPLAY, HOME, COLUMNS, EDITOR, SHELL, ERASE, PATH, and MANPATH. Scripts start applications and variables are set and used in the application.

Neat Stuff

Here is just some neat stuff you can do:

Banner Hello will list a banner in '#'s that can be used with a script.

Clear – clears the terminal

Error Redirection – commands that send errors to the screen can have the output to stderr redirected. If the command is typed incorrectly, the output will go to a location specified. An example can be found in the use of the find command as in:

```
'find / -name timesh -print'
```

When this is executed from the user login many messages like:

```
"find: cannot open /var/sam/lp"
```

will result because your user does not have access to many subdirectories. This will bury your result in the error messages. However, if you add the '2>/dev/null' the error will be sent to the bit bucket as in:

```
'find / -name timesh -print 2>/dev/null'
```

As a result you will only see the result. Errors can be sent to a file as in:

```
'find / -name unxlink2 -print 2>find.err'
```

A '2>>find.err' will append the errors to the find.err file.

Wc – can count words.

As an example:

```
$ wc today
```

```
4 8 50 today
```

The '-l' entry will modify the word count to count the lines as in:

```
$ ls -al | wc -l
```

```
67
```

Cal – or Calendar command will show the calendar.

Find the difference between two files:
'bdiff file1 file2'

Conclusion

The new HP-UX user coming to the platform from MPE/iX can have a very positive experience when moving to HP-UX. One caveat I would note is that UNIX has more complexity in it simply because of the many hands that have been involved in its development. It has many ways to do things and those vary from the shell you are using to the UNIX platform. HP-UX users are more accepting of questions realizing that not any one person can get their arms around all of the information. The new users will note more books on the shelf of an HP-UX user. There will be more working together on problems and more consulting of manuals than I have seen in MPE/iX. Basically, once you achieved a certain level of knowledge you could often finger out sticky problems in MPE/iX by reading HELP, and looking around. HP-UX, takes longer to arrive at that level and one may never get to the same grasp they thought they had of MPE/iX. My suggestion is to do your best to cover what you can learn and then keep your mind open to learn new things and new ways of doing things on a daily basis. Good luck and have fun!!

References

HP Documentation Archive:
<http://docs.hp.com>

Robelle Documentation:
<http://www.robelle.com/library/smugbook/mpe2unix.html>

This link leads to the HP e3000 Transition Guide
http://www.hp.com/products1/mpeixservers/future/quick_ref.html

This link leads to the HP E3000 Transition Training
<http://www.education.hp.com/curr-mpe-e3000.htm>

This link leads to a resource page in the Kubler Consulting, Inc. website the holds several pertinent articles:

<http://www.kublerconsulting.com/mpetounix/UnixCommandsfortheMPEUser>

<http://www.kublerconsulting.com/hpuxperformance/hpuxperformancearticle>

<http://www.kublerconsulting.com/mpetounix/trainingoutline>