Interex HP3000 Madrid Conference, March 1986
Theme: 'Migration to 2001'

. . . universal software in 2001 . . . some UNIphiles would say that the year 2001 will see one world, under UNIX™, indivisible, with liberty and c shells for all . . . here's a light-weight introduction to conversational UNIX™ so you can get food, directions and maybe lodgings for the night in the Migration to 2001 . . . .

UNIX* thru the Eyes of MPE

Sam Boles, Member Technical Staff

HEWLETT PACKARD

With UNIX™ evolving as the de facto "industry standard" operating system, Hewlett-Packard now includes this important dimension in its array of computer technology. The HP9000 computer family currently supports HP-UX, a powerful dialect of UNIX™, with more under development.

Here you can get a view of UNIX™ thru the eyes of MPE. The friendly vernacular of MPE — second language to HP3000 users around the world — becomes the familiar basis in terms of which those new to UNIX™ can acquaint themselves with the terse power of this operating system.

Starting with fundamentals that map one-to-one, you'll see some MPE UDC's used with HP summer students to accelerate their productivity by providing a transitional mechanism from their UNIX™ background. From there you'll move into the more complex facilities that the UNIX™ productivity engine gives to both programmers and end-users alike.

. . . since 1969 . . . a little UNIX™ lore, mystique and culture . . . .

Archeological evidence of UNIX™ dates back to 1969. For the children among us, that's the year one of the leaders in the computer industry "***unbundled***" in recognition of the fact that software was no longer just the packing that came free with the hardware to keep it from rattling around in the carton. Anything that's survived in this business since 1969 has some ***lore, mystique and culture***. UNIX™ is no exception.

First of all, it's an operating system ***of the programmers, for the programmers, by the programmers***. That's why we ***byte hacks love it***. And that's why ***civilized people spend large sums of money for*** commercial ***shells to cover*** the lean terse power of UNIX™.

**UNIX™ is a trademark of AT&T Bell Laboratories.*

Some say it's ***unfriendly***. But you have to remember that one man's "friendly" may be another man's "verbose."

It's the ***classic controversy of efficiency vs friendliness***. Do you have an elaborate high-overhead ritual to establish friendliness or do you get right to the point? Do you ask for a confirmation that the user really wants to purge every file in his group, or do you assume that if he's smart enough to ask for it you'll do it for/to him?

You can look at it this way: a lot of development nodes save trees by running OUTFENCE high, going to hardcopy on only a small portion of printer output. That means to get hardcopy you need to enter something like

```
ALTSPOOLFILE #O936;PRI=10
```

Now that's ***mnemonic and intuitive***, probably. It's maybe what you'd call ***friendly***. But about the third time you do it, you decide it's worth the trouble of updating your UDC's with something like

```
ASF SPL=0,PRI=1,COPIES=1
OPTION LIST
ALTSPOOLFILE #O!SPL;PRI=!PRI;COPIES=!COPIES
```

that you invoke with something like

```
ASF 936 10
```

That's the UNIX™ style.

It's ***terse***.

Crisp.

All right, ***cryptic***.

Powerful.

A rich repertoire of commands and options to do the kinds of things programmers do to build and document software.

. . . maybe something of a misnomer: today, there's little UNI in UNIX™ beyond UNIfying . . . .

UNIX™ may be a ***misnomer***. Legend has it that when the brilliant Ken Thompson named his brilliant child, he did it in counterpoint to the multi-tasking multi-user MULTICS at MIT. His was a single-user system for a single-engineer work station. Today there is ***little UNI*** in UNIX™ beyond the fact that it may be the single most ***UNI***fying element across the wide variety of hardware architectures and configurations in the industry today. Beyond that great ***UNI***fying attribute and signal contribution, UNIX™ is ***MULTItasking, MULTI-user, MULTI-noded, MULTI-shelled, even MULTI-processed*** on the HP9000/500 with its tri-CPU design.

One aspect of the "non-UNI" of UNIX™ is its ***multiplicity of dialects***. This is probably good and bad at the same time. Like any worthwhile software system, UNIX™ is evolving. It's inevitable that such a magnificent theme have myriad variations. The Bell Labs UNIX™, perhaps the seat of orthodoxy, has a System III and a System V. There's the Berkeley 4.1. And HP-UX, XENIX, VENIX, QNX, UNI-plus, -star, -plex and, of course, the powerful NIX of the anti-UNIX™ clingons.

Now we can't hold back tomorrow. We don't even want to. Just ***don't be deluded*** into thinking that the UNIX™ "industry standard" is going to get you entirely out of the ***technological retread business*** we've been confronted with for generations (computer, that is), with all its learning curve entropy and proactive inhibitions.

As Churchill once said about Democracy: ***It's not perfect by any means; it's just the best we've been able to come up with***. The same assessment might apply to UNIX™.

... UNIX™ and MPE side by side, going thru a few ordinary everyday commands ....

First a few words about the examples you see here. We all know the frustration of the ***example that doesn't work***. One way to reduce that problem is to capture the example right at the screen ***in vivo***.

Now to do this for the MPE part is a fairly straightforward exercise if you have an old 2647 like mine. You work the example on the terminal on-line to the computer, then position the cursor at the start of the example. You go into local command mode to do a

```
COPY ALL FROM DISPLAY TO LEFT TAPE
```

When you've gotten the latest batch of examples on tape, you do a

```
MARK LEFT TAPE

REWIND LEFT TAPE
```

Then you get into TDP (Text & Document Processor), find the place where you want to splice in the example, do an

```
/A nnn.nn
```

Then when you get the line number prompt, touch the READ key to get the cartridge tape contents spliced into your text file.

Getting the UNIX™ examples is a little different.

If you're using an HP9000/520, you've got an integrated 5" floppy disc drive. As you do the examples you precede the example with an `echo` or a `cat >>` to the disc file where you're collecting your actual examples. For example, for the `ps` (process show) command:

```
echo $ ps -e >> seb
```

(The `>>` means to append or concatenate the string after `echo` to the target file `seb`.) Then you actually do the command but redirect the output to the same file:

```
ps -e >> seb
```

This gives you what would have been on the screen in your disc file. Then, if you haven't bothered to engineer any better datacom, you can

```
lifcp seb /dev/rfd:SEB
```

to get your examples onto a floppy in LIF (Logical Interchange Format), take it over to your HP9000/236, do a virtual terminal file transfer to the HP3000 where the main body of your text is for doing your laser typesetting via TDP, and join the examples into the appropriate spot.

In the examples you see the HP-UX form, then the MPE form with an HP-UX-like UDC. In the UDC there's an `option list` to show you how the UDC gets expanded and executed. Imagine a Carriage/Cursor Return at the end of each line unless specified otherwise. If there's a ***Control-D*** you'll see `[ctl-D]`.

So much for the logistics. Let's get started.

First, get on the system:

HP-UX:

```
login: boles

Welcome to Hewlett-Packard System 9000 HP-UX
```

MPE:

```
:hello boles.cad
HP3000 / MPE V  G.B0.00 (BASE G.B0.00).  MON, DEC 24, 1984,  4:24 PM
```

Accounting in HP-UX is done generally at the ***user*** level, as opposed to ***user.account*** in MPE. There are some other differences, too. For example, if you have a password and key it wrong, MPE asks you several times to try again; HP-UX doesn't tell you whether it's the user or the password or a backspace that's the problem, but asks for everything again.

HP-UX:

```
$ who am i
boles    console Dec 24 13:26
```

MPE:

```
:whoami
SHOWME
USER: #S85,BOLES.CAD,UNIX       (NOT IN BREAK)
MPE VERSION: HP32033G.B0.00.  (BASE G.B0.00).
CURRENT: MON, DEC 24, 1984,  4:26 PM
LOGON:   MON, DEC 24, 1984,  4:24 PM
CPU SECONDS: 5          CONNECT MINUTES: 2
$STDIN LDEV: 22          $STDLIST LDEV: 22
```

Notice the blanks are suppressed in the UDC to get the `showme`.

HP-UX:

```
$ date
Mon Dec 24 13:48:48 PST 1984
```

MPE:

```
:date
SHOWTIME
MON, DEC 24, 1984,  4:26 PM
```

Basically the same but a little more time granularity in HP-UX and a GMT (Greenwich Mean Time) basis.

HP-UX:

```
$ ps -e
   PID TTY  TIME COMMAND
 27335  co  0:00 ps
 27279  co  0:04 sh
    35   ?  0:01 getty
    34  a2  0:02 getty
    33  a1  0:01 getty
    32  a0  0:01 getty
     1   ?  0:01 init
```

MPE:

```
:pse
SHOWJOB

JOBNUM  STATE IPRI JIN  JLIST    INTRODUCED  JOB NAME

#S69    EXEC        20  20       FRI  8:54A  OPERATOR.SYS
#S85    EXEC        22  22       MON  4:24P  BOLES.CAD

2 JOBS:
    0 INTRO
    0 WAIT; INCL 0 DEFERRED
    2 EXEC; INCL 2 SESSIONS
    0 SUSP
JOBFENCE= 0; JLIMIT= 5; SLIMIT= 60
```

The ps, like showjob, tells you what's running in the system. Some differences are cosmetic: syntax, format, nomenclature; but CPU consumption, state and start time are all useful but not available in both systems with these comparable commands.

HP-UX:

```
$ ls
seb
sebb
```

MPE:

```
:ls
LISTF @

FILENAME

SEBUNX
```

Again, mostly cosmetic differences.

HP-UX:

```
$ ll
total 3
-rw-rw-rw-   1 boles    101          370 Dec 24 13:51 seb
-rw-rw-rw-   1 boles    101          100 Dec 24 13:48 sebb
-rw-rw-rw-   1 boles    101          365 Dec 24 13:51 sebc
```

MPE:

```
:ll
LISTF @,2
ACCOUNT=  CAD          GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

SEBUNX  *          72B  FA          48         48   7       16  1  1
```

The HP-UX "long" file list gives security, date and owner. The *-rw-rw-rw-* means it's an ordinary data file (not a directory nor a device special file), the owner of the file has read and write access but not execute permission, as do the owner's group and the public in general. The listing also includes number of directory links, owner, group code, size in bytes, date and time of last modification.

This touches on a major difference: the file system. The UNIX™ directory structure and file concepts are a major transitional consideration, and beyond the scope of this paper. You get glimpses here in the links information and in the `mkdir` ***and*** `cd` ***examples below. But remember this is only the tip — there's a real iceberg there.***

HP-UX:

```
$ cat
This is to show cat with no parms.
This is to show cat with no parms.
This is line 2 of show cat.
This is line 2 of show cat.
[ctl-D]
```

MPE:

```
:cat
FCOPY FROM=;TO=
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983


This is to show cat with no parms.
This is to show cat with no parms.
This is line 2 of show cat.
This is line 2 of show cat.
 < CONTROL Y >

2 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

This is the `cat` (for con***cat***enate) form without parameters. It's basically input from $STDIN and output to $STDLIST -- the CRT in this case.

HP-UX:

```
$ cat > file1
This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
[ctl-D]
```

MPE:

```
:catt file1
FCOPY FROM=;TO=file1;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
 < CONTROL Y >

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

This is the concatenation of CRT input to a new or replaced file on disc. An easy way to build a file without getting into the editor -- but you give up the more powerful edits. Notice the >. That's UNIX™ redirection from the default CRT to the named file. Be careful: UNIX™ has high regard for your presence of mind. If it finds a file out there already by that name, it doesn't ask as MPE does whether you're sure you want to purge it (unless you've removed the write permission with a chmod) -- it just writes over the old file.

HP-UX:

```
$ cat file1
This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
```

MPE:

```
:catf file1
FCOPY FROM=file1;TO=
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983


This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
EOF FOUND IN FROMFILE AFTER RECORD 2

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

Here with an implicit < redirection of input, we concatenate from the named file to the CRT.

HP-UX:

```
$ cp file1 file2
$ ll file*
-rw-rw-rw-   1 boles    101          121 Dec 24 14:01 file1
-rw-rw-rw-   1 boles    101          121 Dec 25 20:44 file2
$ cat file2
This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
```

MPE:

```
:cp file1 file2
FCOPY FROM=file1;TO=file2;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 2

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:ll file@
LISTF file@,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE1             80B   FA           3       1023   1      128  1  8
FILE2             80B   FA           3       1023   1      128  1  8


:catf file2
FCOPY FROM=file2;TO=
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983


This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
EOF FOUND IN FROMFILE AFTER RECORD 2

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

This is a simple file copy with no changes as you can see from the `ll` and `cat` listings. Note the wild card * that gives you all files starting with "file".

HP-UX:

```
$ cat >> file2
This is some more text to illustrate the
concatenation facility of UNIX in this game
of "Follow the Leader" with MPE.
[ctl-D]
```

MPE:

```
:cattt file2
FILE file2,OLD;ACC=APPEND
FCOPY FROM=;TO=*file2
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

This is some more text to illustrate the
concatenation facility of UNIX in this game
of "Follow the Leader" with MPE.
 < CONTROL Y >

3 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:catf file2
FCOPY FROM=file2;TO=
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983


This is to show the translation of
the UNIX vernacular to an MPE environment.
It's getting harder.
This is some more text to illustrate the
concatenation facility of UNIX in this game
of "Follow the Leader" with MPE.
EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

Here you see a concatenation with the append redirection instead of the replace. Control-D signals End of Data.

HP-UX:

```
$ cp file2 file3
$ cp file2 file3b
$ cp file2 file3c
$ ll file3*
-rw-rw-rw-   1 boles   101          247 Dec 25 20:55 file3
-rw-rw-rw-   1 boles   101          247 Dec 25 20:55 file3b
-rw-rw-rw-   1 boles   101          247 Dec 25 20:55 file3c
$ rm file3*
$ ll file3*
file3* not found
```

MPE:

```
:cp file2 file3
FCOPY FROM=file2;TO=file3;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:cp file2 file3b
FCOPY FROM=file2;TO=file3b;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:cp file2 file3c
FCOPY FROM=file2;TO=file3c;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
:ll file3@
LISTF file3@,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE3             80B   FA           6       1023   1      128  1  8
FILE3B            80B   FA           6       1023   1      128  1  8
FILE3C            80B   FA           6       1023   1      128  1  8
:rm file3
PURGE file3
:rm file3b
PURGE file3b
:rm file3c
PURGE file3c
:ll file3@
LISTF file3@,2
NO FILES FOUND IN FILE-SET (CIWARN 431)
```

Note here the generic purge, representative of the UNIX™ respect for the programmer's presence of mind. (Some of us who only marginally deserve that respect do a lot more back-ups under UNIX™.) You can protect yourself on sensitive files by removing write permission and thereby getting UNIX™ to prompt for confirmation of purge.

HP-UX:

```
$ ll file1
-rw-rw-rw-   1 boles    101          121 Dec 24 14:01 file1
$ chmod 600 file1
-rw-------   1 boles    101          121 Dec 24 14:01 file1
```

MPE:

```
:ll file1
LISTF file1,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE1              80B  FA           3       1023   1      128  1  8


:chmod600 file1
SECURE file1
```

Here's an example of changing access permissions on a file. We don't have a simple parallel in MPE. This now disallows group and public users to access the file. Note that the UNIX™ granularity of control could be approximated by a combination of the `secure` you see here and the `altgroup xxx; access=` facilities in MPE.

HP-UX:

```
$ cp file2 file3
$ ll file*
-rw-------   1 boles    101          121 Dec 24 14:01 file1
-rw-rw-rw-   1 boles    101          247 Dec 25 20:53 file2
-rw-rw-rw-   1 boles    101          247 Dec 25 21:06 file3
$ mv file3 file4
$ ll file*
-rw-------   1 boles    101          121 Dec 24 14:01 file1
-rw-rw-rw-   1 boles    101          247 Dec 25 20:53 file2
-rw-rw-rw-   1 boles    101          247 Dec 25 21:06 file4
```

MPE:

```
:cp file2 file3
FCOPY FROM=file2;TO=file3;NEW
HP32212A.3.18 FILE COPIER (C) HEWLETT-PACKARD CO. 1983

EOF FOUND IN FROMFILE AFTER RECORD 5

6 RECORDS PROCESSED *** 0 ERRORS


END OF SUBSYSTEM
```

```
:ll file@
LISTF file@,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE1             80B   FA           3       1023   1      128  1  8
FILE2             80B   FA           6       1023   1      128  1  8
FILE3             80B   FA           6       1023   1      128  1  8


:mv file3 file4
RENAME file3,file4
:ll file@
LISTF file@,2
ACCOUNT=  CAD         GROUP=  UNIX

FILENAME  CODE  ------------LOGICAL RECORD-----------  ----SPACE----
                  SIZE  TYP        EOF      LIMIT R/B  SECTORS #X MX

FILE1             80B   FA           3       1023   1      128  1  8
FILE2             80B   FA           6       1023   1      128  1  8
FILE4             80B   FA           6       1023   1      128  1  8
```

Here you see the *move* or rename facility in action.

HP-UX:

```
$ pwd
/users/boles
$ mkdir dir2
$ ll
total 15
drwxrwxrwx   1 boles    101          0 Dec 25 21:26 dir2
-rw-------   1 boles    101        121 Dec 24 14:01 file1
-rw-rw-rw-   1 boles    101        247 Dec 25 20:53 file2
-rw-rw-rw-   1 boles    101        247 Dec 25 21:06 file4
-rw-rw-rw-   1 boles    101       2536 Dec 25 21:28 seb
-rw-rw-rw-   1 boles    101       1055 Dec 24 14:07 sebe
-rw-rw-rw-   1 boles    101       1055 Dec 25 20:43 sebf
-rw-rw-rw-   1 boles    101       1607 Dec 25 20:54 sebg
-rw-rw-rw-   1 boles    101       2059 Dec 25 21:04 sebh
$ cd dir2
$ pwd
/users/boles/dir2
$ cp ../file1 file1dir2
$ ll
total 1
-rw-------   1 boles    101        121 Dec 25 21:35 file1dir2
$ cd
$ pwd
/users/boles
```

MPE:

Here we don't have an MPE analog closer than `hello` with a new group specified. In the example, while in the home directory, you ***make*** a new ***dir***ectory with `mkdir`. The directory file (initial "d" in the `ll` listing) now appears in its parent directory. A `cd` (***c***hange ***d***irectory) to the subdirectory `dir2` is confirmed with a `pwd` showing the path name up the directory chain. The `cp` uses a `../` to indicate the parent directory of the current working directory. A `cd` without an explicit directory gets us back to the home directory, which the `pwd` confirms. This is just a quick dip in the deep end of the pool. Don't worry about this for the brief glimpse of UNIX™ you get here. But do be aware that the UNIX™ file system is different from MPE.

```
$ who > file4
$ cat file4
boles    console Dec 24 13:26
$ date >> file4
$ cat file4
boles    console Dec 24 13:26
Tue Dec 25 21:42:10 PST 1984
$ ll file4
-rw-rw-rw-   1 boles    101          59 Dec 25 21:42 file4
$ wc file4
      2     11     59 file4
$ grep 'MPE' file1
the UNIX vernacular to an MPE environment.
$ ll file1 > temp1; wc file1 > temp2; cat temp1 temp2 | grep 'file'
-rw-------   1 boles    101         121 Dec 24 14:01 file1
      5     22    121 file1
```

MPE:

Don't try to map this one-for-one to MPE. You see some redirection and a new counter command, `wc`, to set up an illustration of ***piping*** (the | operand) and the string finder, `grep`. The `wc` counts lines, words and characters. The `grep` lists the lines that contain the ***g***eneral ***e***x***p***ression ("`mess`": ***m***nemonics are merely a ***s***tate of mind) search string argument. Here the `cat` has 2 input files that it ***pipes*** to `grep` which outputs the two lines containing the search string 'file'.

Before wrapping up, let's scratch the surface of the UNIX™ ***shell***. First some simple shell scripts. Suppose you want your UNIX™ to speak MPE. Here you see a file called `listf` that contains `ls`. At first it won't execute but the `chmod` fixes that.

```
$ cat > listf
ls
[ctl-D]
$ listf
listf: cannot execute
$ chmod 777 listf
$ listf
dir2    file2 . . . temp1   temp3
file1   file4 . . . temp2
```

Here you see a file called `purge` with a `$1`, the symbol for the first argument, which is the name of the file you want to purge. The `chmod` makes it executable.

```
$ cat > purge
rm $1
[ctl-D]
$ chmod 777 purge
$ purge temp4
```

Next you see a shell control loop that edits all the files starting with "file", using the commands in `sebmod`.

```
$ cat sebsh
for i in file*
  do ed $i < sebmod
  done
```

Here's what `sebmod` looks like. It gives the file name, lists the first record, inserts a new line, then lists the first three lines, then quits.

```
$ cat sebmod
f
1
i
Begin with Mar 1986 Interex HP3000 Conf . . .
.
1,3p
q!
```

Here's an execution:

```
$ sebsh
107
file1
This is to show the translation of
Begin with Mar 1986 Interex HP3000 Conf . . .
This is to show the translation of
the UNIX vernacular to an MPE environment.

233
file2
This is to show the translation of
Begin with Mar 1986 Interex HP3000 Conf . . .
This is to show the translation of
the UNIX vernacular to an MPE environment.

59
file4
boles    console Dec 24 13:26
Begin with Mar 1986 Interex HP3000 Conf . . .
boles    console Dec 24 13:26
Tue Dec 25 21:42:10 PST 1984
```

Epilogue . . .

There you have it: a glimpse thru the Looking Glass from MPE-land into the Land of UNIX™. You've seen our "MPENIX": some of the elementary functions in our quasi-UNIX™ UDC's that we built to help our summer students. From there you sampled some of the UNIX power that enables a computer user to reach new levels of productivity. You've seen some of the features that have enabled UNIX™ to establish a good track record as the common link that lets us move with reasonable gracefulness across a substantial portion of the computer world today.

About the Author ....

Sam Boles *is a Member Technical Staff in the Hewlett-Packard Information Software Operation in Cupertino, California. With HP since 1976, his computer experience started back in the AUTOCODER days of the 1401/1410, migrated thru the 360/370 era, and now focuses on next-generation operating system software. Sam received his MS at UCLA in Information Systems.*

sebiug48 2135 27jan86