SCIENCE AND ENGINEERING RESEARCH COUNCIL
RUTHERFORD APPLETON LABORATORY

COMPUTING DIVISION

DISTRIBUTED COMPUTING NOTE 554

PERQ UNIX IMPLEMENTATION NOTE #7
Unix overall design

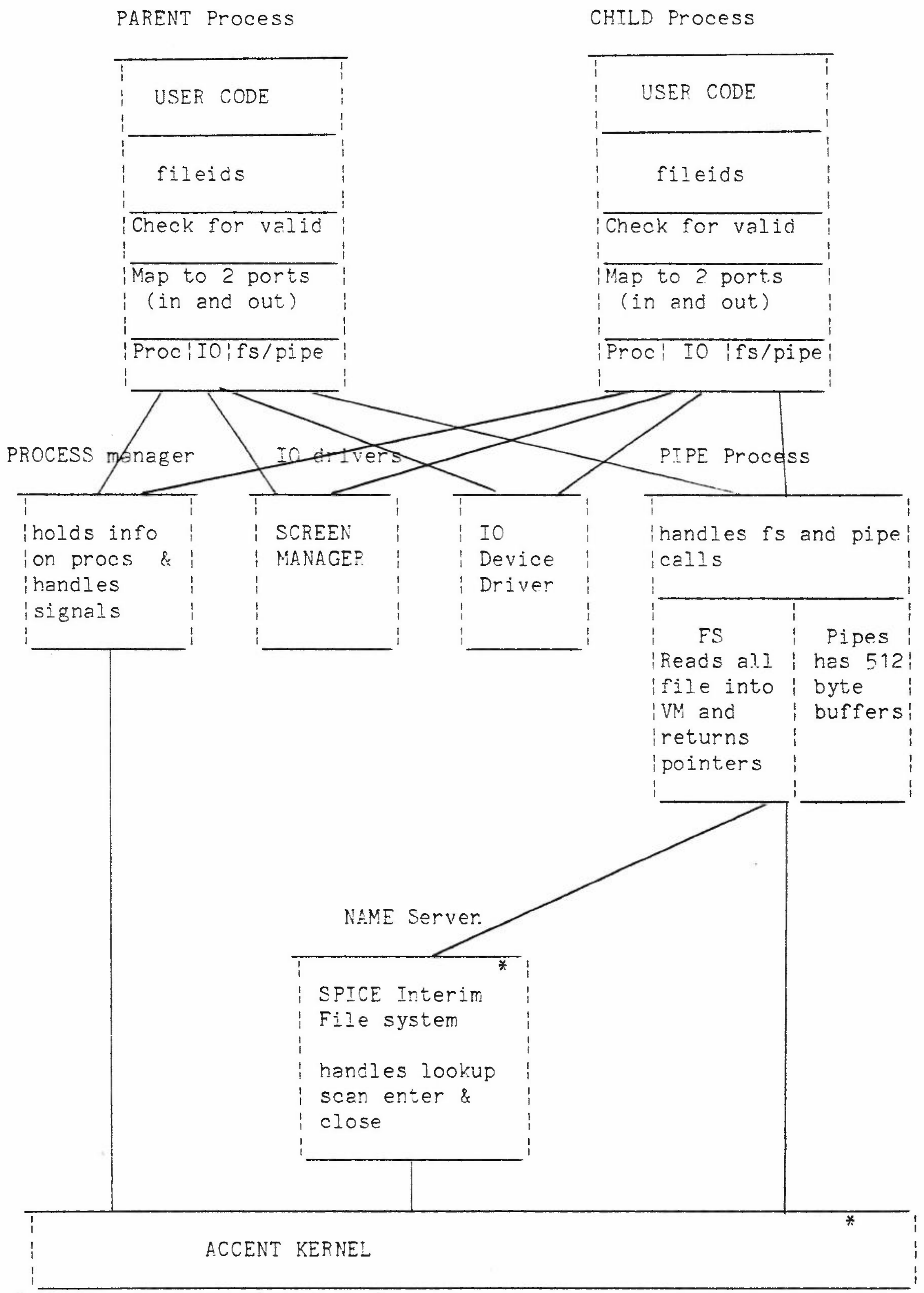
Issued by
L O Ford

17 February 1982

DISTRIBUTION:

R W Witty
C Prosser
A S Williams
L O Ford
T Watson
E V C Fielding
F R A Hopgood
J C Malone
ICL
RL Support/PERQ/Unix Implementation Notes file

This paper attempts to outline the design for implementing UNIX on top of ACCENT. The first diagram shows the various processes which exist and those which need to be written. The interconnections between the processes are also shown.



* denotes process already exists

The PIPE process which handles reads and writes to files or pipes is necessary to support multiple readers and writers on both pipes and files as the SPICE interim filesystem does not support these facilities.

It was decided that both of these functions should reside in the same process as we may need to support named pipes in future. This requires that the pipe process interacts with the name server.

An open on a physical device will be handled initially by PIPE and the name server and the appropriate ports for the device driver will be returned to the user by PIPE.

Mappings of Unix system calls to ACCENT

Unix Call	Spice FS/ACCENT	Notes
ACCESS - determine accessibility of a file	AFSLookup	NS Server
ACCT - turn accounting on or off	-	not implemented
ALARM - schedule signal after specified time	-	handled in Process Manager -PROCM will send signal after specified time
BRK,SBRK,BREAK - change core allocation	ValidateMemory	-
CHDIR,CHROOT - change default directory	AFSetPath	(possibly + AFSGetPath)
CHMOD - change mode of file	AFSLookup	does copy on disk get changed if write into FSDataentry
CHOWN (SU) - change owner and group of a file	AFSLookup	"
CLOSE - close a file (or pipe)	AFSClose (for file)	PIPE will handle pipe - An in-use count will be decremented and the file closed only when the last process has closed the file or pipe. For a file WRITESPICESEGMENT must be called to update the file on disk.
CREAT - create a new file	AFSCreate	UNIX allows truncation of an existing file ACCENT only allows creation of a new file PIPE must check on a creat to see if file already exists

DUP,DUP2 - duplicate an open file descriptor	-	handled in User backend
EXECL etc - execute a file	-	will need to implement LOADER.PAS The exec call will overlay the user code in the processes VM
EXIT - terminate process	Terminate	PIPE needs to close files
FORK - spawn new process	Fork	PIPE needs to duplicate file descriptors and the child must inform process manager of its kernelport
GETPID - get process identification	-	return Kernelport
GETUID, GETGID, GETEUID, GETEGID - get user and group identity	-	handled by process manager
IOCTL,STTY,GTTY - control device	-	need to be defined for PERQ devices
KILL - send signal to a process	Terminate	handled by Process manager
LINK - link to a file	-	no links in interim filesystem - possibly implement a form of symbolic link
LOCK - lock a process in primary memory	-	NOP
LSEEK,TELL - move read/write pointer	-	handled by PIPE
MKNOD (SU) - make a directory or special file	?	create file with extension
MOUNT,UMOUNT - mount or remove file system	AFSPMount,AFSPDisMount	-

MPX - create and manipulate multiplexed files	-	NOP
NICE - set program priority	SetPriority	-
OPEN - open for reading or writing	AFSOpen	An open request must check the list of files already open (held in PIPE) and if already open increment an in-use count. PIPE must also determine if the open is for a physical device and if so return the appropriate process ports. The name server is called for both files and physical devices. A bit in the file header will be used to indicate a physical device. The whole file is mapped into the VM of the PIPE process (READSPICESEGMENT) for an open on a file.
PAUSE - stop until signal	-	handled by Process manager
PHYS (SU) - allows a process to access physical addresses	-	NOP
PIPE - create an interprocess channel	-	handled by PIPE - a check is made to see if the pipe is already open. If it is not a 512 byte buffer is allocated.
PKON,PKOFF - establish packet protocol	-	NOP
PROFIL - execution time profile	-	NOP
PTRACE - process trace	-	NOP

READ - read from a file (or a pipe)	-	read requests are passed to the appropriate process by the user backend code (PIPE for fs and pipes, the appropriate IO driver for physical devices).
SETUID, SETGID - set user and group ID	-	handled by process manager
SIGNAL - catch or ignore signals	-	handled by User backend and PROC using exception mechanism
STAT, FSTAT - get file status	AFSLookup	-
STIME (SU) - set time	-	Process manager
SYNC - update super-block	-	NOP
TIME, FTIME - get time and date	-	Process manager
TIMES - get process times	-	Process manager (elapsed)
UMASK - set file creation mode mask	-	handled by PIPE
UNLINK - remove directory entry	AFSDelete	-
UTIME - set file times	AFSLookup	-
WAIT - wait for process to terminate	-	handled by PIPE
WRITE - write on a file (or a pipe)	-	The write request is passed to the appropriate process by the user backend code (PIPE for fs and pipes, IO drivers for physical devices).

Mappings of Unix system calls to processes

PIPE

chdir
close
create
fork
lseek/tell
mknode
mount
open
pipe
read
umask
unlink
utime
write

PROCESS MANAGER

alarm
exit
fork
getpid
kill
pause
settime
times
wait

IO drivers

close
ioctl
lseek/tell
read
write

USER backend

dup
exec
sbrk
signal

NOP

access
acct
chmod
chown
getgid
getuid
geteuid
getegid
link
lock
mpx
nice
phys
profil
ptrace
sync