Relación de problemas. Sistemas Operativos II

Tema 1: Implementación de los sistemas de archivos

1. Sea un Sistema Operativo que sólo soporta un directorio (es decir, todos los archivos existentes estarán al mismo nivel), pero permite que los nombres de archivo sean de longitud variable. Apoyándonos únicamente en los servicios proporcionados por este Sistema Operativo, deseamos construir una "utilidad" que "simule" un sistema jerárquico de archivos. ¿Es esto posible? ¿Cómo?

2. En un entorno multiusuario, cada usuario tiene un directorio inicial al entrar en el sistema a partir del cual puede crear archivos y subdirectorios. Surge, entonces, la necesidad de limitar el tamaño de este directorio para impedir que el usuario consuma un espacio de disco excesivo. ¿De qué forma el Sistema Operativo podría implementar la limitación de tamaño de un directorio?

3. En la siguiente figura se representa una tabla FAT. Al borde de sus entradas se ha escrito, como ayuda de referencia, el número correspondiente al bloque en cuestión. También se ha representado la entrada de cierto directorio. Como simplificación del ejemplo, suponemos que en cada entrada del directorio se almacena: Nombre de archivo/directorio, el tipo (F=archivo, D=directorio), la fecha de creación y el número del bloque inicial.

Nombre	Tipo	Fecha	NºBloque
DATOS	F	8-2-90	3

FAT

1		10	
2		11	
3	15	12	
4		13	
5		14	
6		15	*
7		16	
8		17	
9		18	

Tenga en cuenta que:
- el tamaño de bloque es de 512 bytes
- el asterisco indica último bloque
- todo lo que está en blanco en la figura está libre.

Rellene la figura para representar lo siguiente:

a) Creación del archivo DATOS1 con fecha 1-3-90, y tamaño de 10 bytes.
b) Creación del archivo DATOS2 con fecha 2-3-90, y tamaño 1200 bytes.
c) El archivo DATOS aumenta de tamaño, necesitando 2 bloques más.
d) Creación del directorio D, con fecha 3-3-90, y tamaño 1 bloque.
e) Creación del archivo CARTAS con fecha 13-3-90 y tamaño 2 kBytes.

4. Si usamos un Mapa de Bits para la gestión del espacio libre, especifique la sucesión de bits que contendría respecto a los 18 bloques del ejercicio anterior.

5. Si se pierde el primer puntero de la lista de espacio libre, ¿podría el Sistema Operativo reconstruirla? ¿Cómo?

6. El espacio libre en disco puede ser implementado usando una lista encadenada con agrupación o un mapa de bits. La dirección en disco requiere D bits. Sea un disco con B bloques, en que F están libres. ¿En qué condición la lista usa menos espacio que el mapa de bits?

7. Entre los posibles atributos de un archivo, existe un bit que marca un archivo como temporal y por lo tanto esta sujeto a destrucción automática cuando el proceso acaba ¿Cuál es la razón de esto? Después de todo un proceso siempre puede destruir sus archivos, si así lo decide.

8. Algunos SO proporcionan una llamada al sistema (RENAME) para dar un nombre nuevo a un archivo existente ¿Existe alguna diferencia entre utilizar esta llamada para renombrar un archivo y copiar el archivo a uno nuevo, con el nuevo nombre y destruyendo el antiguo?

9. Un i-nodo de UNIX tiene 10 direcciones de disco para los diez primeros bloques de datos, y tres direcciones más para realizar una indexación a uno, dos y tres niveles. Si cada bloque índice tiene 256 direcciones de bloques de disco, cuál es el tamaño del mayor archivo que puede ser manejado, suponiendo que 1 bloque de disco es de 1KByte?

10. Sobre conversión de direcciones lógicas dentro de un archivo a direcciones físicas de disco. Estamos utilizando la estrategia de indexación a tres niveles para asignar espacio en disco. Tenemos que el tamaño de bloque es igual a 512 bytes, y el tamaño de puntero es de 4 bytes. Se recibe la solicitud por parte de un proceso de usuario de leer el carácter número N de determinado archivo. Suponemos que ya hemos leído la entrada del directorio asociada a este archivo, es decir, tenemos en memoria los datos PRIMER-BLOQUE y TAMAÑO. Calcule la sucesión de direcciones de bloque que se leen hasta llegar al bloque de datos que posee el citado carácter.

11. ¿Qué organización de archivos elegiría para maximizar la eficiencia en términos de velocidad de acceso, uso del espacio de almacenamiento y facilidad de modificación (añadir/borrar /modificar), cuando los datos son:

    a) modificados infrecuentemente, y accedidos frecuentemente de forma aleatoria
    b) modificados con frecuencia, y accedidos en su totalidad con cierta frecuencia
    c) modificados frecuentemente y accedidos aleatoriamente y frecuentemente

12. Un controlador en software de disco duro recibe peticiones para los cilindros 10, 22, 20, 2, 40, 6 y 38 en ese orden. El brazo tarda 6ms en moverse de un cilindro al siguiente Suponiendo que el brazo está inicialmente en el cilindro 20 ¿cuánto tiempo de posicionamiento se requiere siguiendo los algoritmos
    a. FCFS ?
    b. SSTF ?
    c. SCAN ?

13. ¿Tendría sentido combinar en un mismo sistema de archivos dos métodos de asignación de espacio en disco diferentes? Por ejemplo, el método contiguo combinado con el enlazado. Ponga un ejemplo y razónelo.

14. ¿Por qué la FAT en MS-DOS y Windows tiene un tamaño fijo en disco para cada sistema de archivos?

15. Para comprobar la consistencia de un sistema de archivos de Unix, el comprobador de consistencia (*fsck*) construye dos listas de contadores (cada contador mantiene información de un bloque de disco), la primera lista registra si el bloque está asignado a algún archivo y la segunda, si está libre. Según se muestra en la siguiente figura:

en uso:	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0
libres:	0	0	0	1	1	1	0	0	0	1	0	1	1	0	1

¿Existen errores? ¿Son serios estos errores? ¿porqué? ¿qué acciones correctivas sería necesario realizar sobre la información del sistema de archivos?

Tema 2: Implementación de un núcleo de un sistema operativo

1. Cuando se abre el archivo /usr/ast/work/f, se necesitan varios accesos a disco. Calcule el número de accesos a disco requeridos (como máximo) bajo la suposición de que el i-nodo raíz ya se encuentra en memoria y que todos los directorios necesitan como máximo 1 bloque para almacenar los datos de sus archivos.

2. ¿El planificador de Unix favorece a los procesos limitados por E/S (cortos) frente a los procesos limitados por CPU (largos)? Explique cómo lo hace. ¿Y el planificador de Windows 2000 /NT?

3. Supongamos que un proceso, **P1,** abre el archivo "**datos**" en modo **lectura/escritura** y otro proceso, **P2**, abre el mismo archivo y con el mismo modo, y a continuación crea un proceso hijo que abre el archivo "**/usr/pepe/doc**" en modo **lectura/escritura**. Represente toda la información relevante sobre el estado de las tablas de descriptores de archivos, tabla de archivos y tabla de i-nodos después de dichas operaciones.

4. ¿Qué pasos se tienen que llevar a cabo en Unix para eliminar **toda** la información de un proceso que ha ejecutado la llamada al sistema *exit*?

5. Supongamos que se permite que dos o más búferes de la caché puedan contener el mismo bloque de disco simultáneamente ¿Qué problemas podrían presentarse? Ponga un ejemplo ¿Cómo lo resuelve el sistema?

6. Suponiendo una ejecución correcta de las siguientes órdenes en el sistema operativo Unix:

```
/home/jgarcia/prog > ls  -i        (* lista los archivos y sus números de i-nodos del directorio prog*)
        18020                 fich1.c
        18071                 fich2.c
        18001                 pract1.c

/home/jgarcia/prog > cd ../tmp
/home/jgarcia/tmp > ln -s ../prog/pract1.c p1.c          (* crea un enlace simbólico *)
/home/jgarcia/tmp > ln ../prog/pract1.c p2.c             (* crea un enlace absoluto o duro*)
```

   represente gráficamente cómo y dónde quedaría reflejada y almacenada **toda** la información referente a la creación anterior de un enlace simbólico y absoluto ("hard") a un mismo archivo, **pract1.c**.

7. Los números de i-nodos de un sistema de archivos en Unix comienzan a numerarse por 1 y no por 0 ¿Cuál es la razón?

8. En Unix, ¿qué espacio total (en bytes) se requiere para almacenar la información sobre la localización física de un archivo que ocupa 3 Mbytes?. Suponga que el tamaño de un bloque lógico es de 1 Kbytes y se utilizan direcciones de 4 bytes. Justifique la solución detalladamente.

9. ¿Por qué no existe una lista enlazada de i-nodos libres (almacenada en disco) que contenga información de todos los i-nodos libres de un sistema de archivos de Unix?

10. ¿Cuál es la razón de que el sistema Unix mantenga una ***Tabla de Regiones*** en vez de almacenar toda la información acerca de las regiones de un proceso en su ***Tabla de Regiones Por Proceso***?

11. ¿Por qué se almacena la **prioridad** de un proceso en su entrada correspondiente de la ***Tabla de Procesos*** y no en su ***u-área***?

12. Supongamos que un proceso va a ejecutar la llamada al sistema ***open*** la cual, a su vez, utiliza el algoritmo de más bajo nivel ***namei*** ¿Qué ocurre en el contexto del proceso a nivel de sistema antes, durante y después de la ejecución de la llamada (céntrese en las estructuras de datos relacionadas con dicho contexto)?

13. Respecto a la gestión de memoria: suponga que un proceso realiza una llamada al sistema ***fork*** creando un proceso hijo. Represente gráficamente como quedan las estructuras de datos relacionadas con ambos procesos (tablas de regiones por proceso, tabla de regiones, tablas de páginas y tabla de marcos de página).

14. ¿Por qué en el sistema de archivos DOS (FAT) está limitado el número de archivos y/o directorios que descienden directamente del directorio raíz y en Unix no lo está?

15. ¿Cuál es el problema que se plantea cuando un proceso no realiza la llamada al sistema ***wait*** para cada uno de sus procesos hijos que han terminado su ejecución? ¿Qué efecto puede producir esto en el sistema?

16. Tenemos un archivo de 4 MBytes, calcule el espacio en disco necesario para almacenar el archivo (incluidos sus metadatos) en un Sistema de Archivos FAT32 y en otro s5fs. Suponga que el tamaño del cluster en FAT32 y el de bloque lógico en s5fs es de 4KBytes.

17. ¿Qué información comparten un proceso y su hijo después de ejecutar el siguiente código? Justifique su respuesta e indique qué hace este trozo de código.

```
if  ( fork() != 0 )
     wait (&status)
else
     exec (B);
```

// usamos una llamada al sistema exec genérica con un único argumento, el nombre de un archivo ejecutable

18. En la mayoría de los sistemas operativos, el modelo para manejar un archivo es el siguiente:
    - Abrir el archivo, que nos devuelve un descriptor de archivo asociado a él.
    - Acceder al archivo a través de ese descriptor devuelto por el sistema.

    ¿Cuáles son las razones de hacerlo así? ¿Por qué no, por ejemplo, se especifica el archivo a manipular en cada operación que se realice sobre él?

19. ¿Para qué se utilizan los algoritmos **bloquear** y **desbloquear** (*sleep* y *wakeup*) en el núcleo de UNIX? Especifique su funcionamiento conjunto para alcanzar su objetivo.

20. ¿Cómo consigue optimizar FFS operaciones del tipo "ls –l" sobre un directorio? Explíquelo y ponga un ejemplo.

21. Sea un directorio cualquiera en un sistema de archivos de Unix, por ejemplo, DirB. De él cuelgan algunos archivos que están en uso por uno o más procesos. ¿Es posible usar este directorio como punto de montaje? Justifíquelo.

22. ¿Es correcto el siguiente programa en UNIX, en el sentido de que no genera inconsistencias en las estructuras de datos que representan el estado de los archivos en el sistema? Represente con un esquema gráfico la información almacenada en las estructuras (tablas de descriptores de archivos, tabla de archivos y tabla de i-nodos) que representan el estado real referente a los archivos utilizados por los siguientes procesos en el instante de tiempo en que: el *PROCESO HIJO DE A* ha ejecutado la instrucción marcada con *(1),* el *PADRE (PROCESO A)* ha ejecutado la *(2)*, y el *PROCESO B* ha ejecutado la *(3)*. Para ello suponga que son los únicos procesos ejecutándose en el sistema en ese momento.

```
/* PROCESO A */
main(){
int archivos[2], pid;
archivos[0]=open("/users/prog.c",O_RDONLY);
archivos[1]=open("/users/prog.c",O_RDWR);
if ((pid=fork())==-1)
perror("Ha ocurrido un error en la creación del proceso);
else if (pid==0)
execlp ("ls", "ls", NULL); /*(1)*/
else
close(archivos[0]); /*(2)*/
}
/* PROCESO B */
main(){
int archivos[2];
archivos[0]=open("/users/prog.c",O_RDONLY);
archivos[1]=open("/users/texto.txt",O_RDWR);
dup (archivos[0]); /* (3) */
}
```

Tema 3: Planificación de recursos. Interbloqueos

1. Considere el siguiente sistema con cinco procesos (P1, P2, P3, P4 y P5) que compiten por cuatro clases de recursos (r1, r2, r3 y r4) y cuyas matrices A y S son:

	r1	r2	r3	r4
P1	0	0	1	2
P2	1	0	0	0
P3	1	3	5	4
P4	0	6	3	2
P5	0	0	1	4

Matriz A (recursos asignados)

	r1	r2	r3	r4
P1	0	0	0	1
P2	0	7	5	0
P3	2	6	0	2
P4	2	0	0	2
P5	2	0	2	0

Matriz S (recursos solicitados)

D = (1 5 2 0)

¿Está el sistema interbloqueado? Utilice el algoritmo de detección para comprobar si lo está o no.

2. Suponga que en el algoritmo del banquero con recursos múltiples se da la relación expresada más abajo, entre los elementos de la matriz de asignación de recursos (A_{ij}), la matriz de peticiones de recursos de los procesos (R_{ij}) y el vector de recursos máximos existentes (E_j). Los índices i, j, se refieren a los procesos y los recursos, respectivamente.

$$A_{ij} + R_{ij} > E_j \quad \text{Para algún valor de i.}$$

¿Qué implicaciones puede tener esto para que todos los procesos acaben sin interbloqueo?

3. A continuación se muestra una secuencia de peticiones de 3 recursos (R, S, T) por parte de dos procesos (A, B) ¿Se ha producido un interbloqueo?

   1. A pide R
   2. B pide T
   3. A pide S
   4. B pide R

   Suponga que el paso 4, B hubiera pedido el recurso S en lugar del R. ¿Hubiera llevado esto a interbloqueo? ¿Y si hubiera pedido los recursos S y R?

4. Un computador tiene 6 unidades de cinta, con *n* procesos compitiendo por ellas. Cada proceso puede necesitar 2 unidades de cinta. ¿ para qué valores de *n* el sistema está libre de interbloqueos?

5. Suponga la siguiente matriz en el algoritmo del banquero:

	Tiene	Max
A	1	6
B	1	5
C	2	4
D	4	7

libres = 2

Si D pide una unidad más, ¿llevaría esto a un estado seguro o inseguro? ¿Qué hubiera pasado si la petición hubiera venido de C en lugar de D?

6. Considere el siguiente estado de un sistema donde existen cinco procesos (A, B, C, D, E) y cuatro recursos (r1, r2, r3, r4):

	r1	r2	r3	r4
A	0	0	1	1
B	2	0	0	0
C	0	0	3	4
D	2	3	5	4
E	0	3	3	2

Matriz A (recursos asignados)

	r1	r2	r3	r4
A	0	0	0	1
B	0	7	5	0
C	6	6	2	2
D	2	0	0	2
E	0	3	2	0

Matriz R (recursos que todavía se necesitan)

E = (6 7 12 12)

D = (2 1 0 1)

a) ¿Está el sistema actualmente en un estado seguro o inseguro?

b) Si C solicita un recurso del tipo r2, ¿cuál sería la respuesta del sistema si sabemos que utiliza el algoritmo del banquero para tomar su decisión?

c) ¿Y si después, C también solicita un recurso del tipo r1?

7. Considere un sistema con ***m*** recursos del mismo tipo, compartidos por ***n*** procesos. Suponga que se dan simultáneamente las siguientes condiciones:
   - Cada recurso es usado por un único proceso en un instante dato. Esto no quiere decir que un proceso no disponga de más de un recurso al mismo tiempo.
   - El máximo número de recursos que necesita cada proceso es ***m***.
   - La suma de las necesidades máximas de todos los procesos es menor que ***m*** + ***n***.

   ¿Está el sistema libre de interbloqueos? Demuéstrelo.