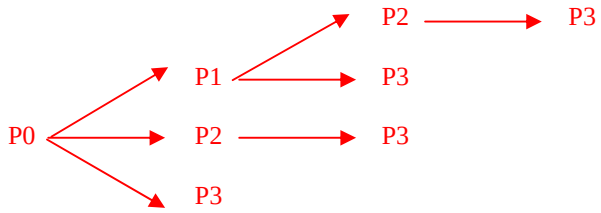
Sistemas Operativos UCM 2011/2012

Soluciones Módulo 2.1 Procesos

1. Dado el siguiente programa …
   a. Dibuje el esquema jerárquico de procesos que se genera para *argc = 3*



   b. ¿Cuántos procesos se crean para *argc* = n? → 2^n

2. Use las llamadas *fork()*, *exec()*, *exit()* y *wait()* de UNIX …

```
P0:
        pidP1 = fork(); → exec("P1")
        wait(pidP1);
        pidP2 = fork()  → exec("P2")
        pidP5 = fork()  → exec("P5")
        pidP7 = fork(); → exec("P7")
        wait(pidP2);
        pidP3 = fork()  → exec("P3")
        pidP4 = fork(); → exec("P4")
        wait(pidP4, pidP5);
        pidP6 = fork()  → exec("P6")
        wait(pidP3, pidP6, pidP7)
        exec("P8")
```

```
P1:
        Código de P1
        exit()
P2:
        Código de P2
        exit()
P3:
        Código de P3
        exit()
etc…
```

Alternativamente …

```
// programa problema2

void T0(){
      {tarea P1}
      envia_señal(S1,S2);
      {tarea P5}
      espera_señal(S3);
      {tarea P6}
      espera_señal(S4,S5);
      {tarea P8}
   }

void T1(){
      espera_señal(S2);
      {tarea P7}
      envia_señal(S5);
   }
```

```
void T2(){
      espera_señal(S1);
      {tarea P2}
      envia_señal(S6);
      {tarea P4}
      envia_señal(S3);
   }

void T3(){
      espera_señal(S6);
      {tarea P3}
      envia_señal(S4);
   }

main() {
   {inicialización}
   cobegin {
      T0;T1;T2;T3
   }
}
```

3. Podemos describir gran parte de la gestión del procesador …

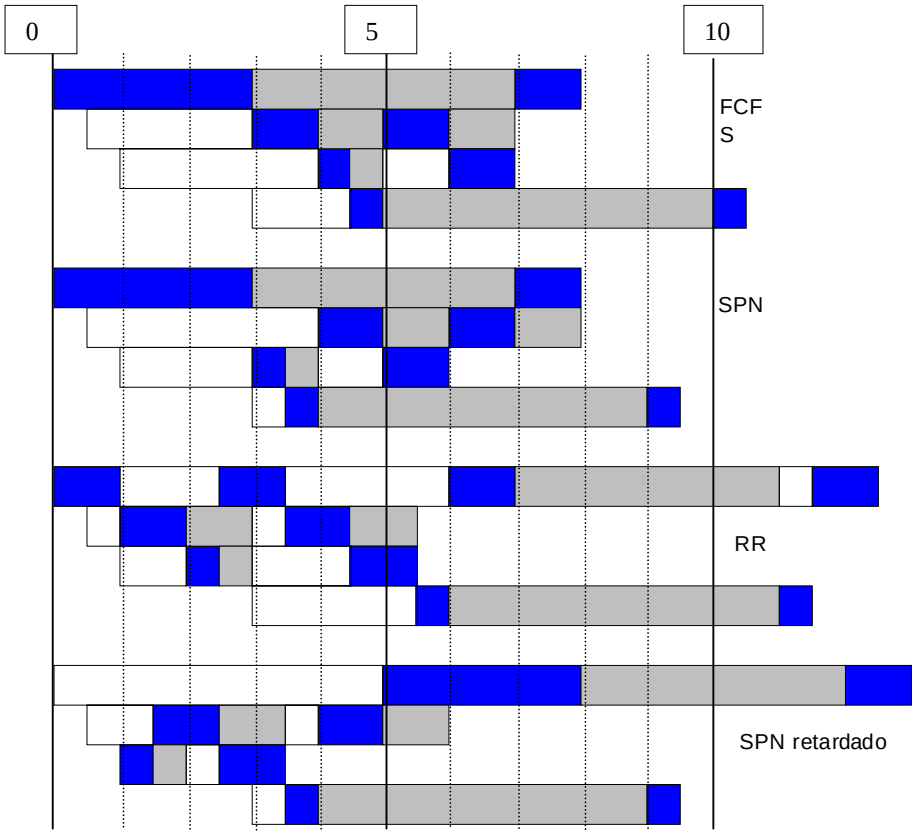
   a) Eventos:
      1. Liberación de la UCP y paso a ejecutar un nuevo proceso
      2. Fin del cuanto de ejecución o expropiación por prioridad.
      3. Llamada a E/S, espera por sincronización o por retardo.
      4. Fin de operación de E/S, o de fin de plazo, o señalización.
   b) Siempre que exista al menos un proceso en la cola de preparados.
   c) 2 → 1: Siempre que existan procesos preparados

      3 → 2: Nunca. En ejecución sólo hay un proceso, que puede realizar una sóla transición.

      4 → 1: Sólo si ambas transiciones las realiza el mismo proceso, para lo cual la UCP debería estar desocupada o ese proceso debería expropiar al que actualmente está en uso de la UCP (lo cual supone una transición 2 adicional). O bien, si la política de planificación empleada recalcula las prioridades (pej.: política HPRN con expropiación) y determina que hay un proceso preparado más prioritario
   d) No hay transición:
      1. Si la UCP está desocupada (es condición previa)
      2. Nunca
      3. Si solicita E/S y no hay procesos preparados para usar la UCP. En realidad en muchos sistemas operativos suele haber un proceso ficticio de mínima prioridad, expropiable por cualquiera y siempre preparado, que se emplea para tratar esta situación, ya que en realidad la UCP no puede estar sin operar (a no ser que se desconecte el sistema) por lo cual esta transición siempre provocaría una transición 1.
      4. Si acaba una operación de E/S y hay un proceso en ejecución no expropiable.

4. Suponga que los siguientes trabajos llegan a procesarse en los instantes indicados

	P1	P2	P3	P4	Medias
FCFS	0	2,5	4	1,5	Tesp = 2
	8	6,5	6	7,5	Prod = 0,381
SPN	0	3,5	3	0,5	Tesp = 1,75
	8	7,5	5	6,5	Prod = 0,421
RR (q=1)	4,5	1,0	2,5	2,5	Tesp = 2,65
	12,5	5,0	4,5	8,5	Prod = 0,32
SPN retardado	5	1,5	0,5	0,5	Tesp = 1,875
	13	6	2,5	6,5	Prod = 0,308



La primera fila de cada entrada muestra el tiempo de espera de cada trabajo para la estrategia considerada y el tiempo medio de espera al final. La segunda fila muestra el tiempo de retorno de cada trabajo y la productividad para la estrategia considerada.

5. Cinco trabajos de lotes, de *A* a *E*, llegan a un centro de cálculo …

a) Inicialmente el tiempo se reparte entre los 5 procesos equitativamente hasta que termina de ejecutarse el más corto, C, trás disponer de sus 2 unidades de tiempo de CPU. En ello se emplean 2 x 5 = 10 unidades de tiempo. Todos los procesos restantes también han consumido 2 unidades de su tiempo de ejecución.
A continuación, se reparte el tiempo entre los 4 procesos restantes, hasta que el siguiente proceso más corto, D, disponga de las 4 -2 = 2 unidades que le restan para terminar. Se emplean en ello 2 x 4 = 8 unidades más.
Razonando de este modo los tiempos de retorno de los diferentes procesos son:

C - 2 x 5 = 10
D - 10 + 2 x 4 = 18 Y el tiempo medio de retorno es
B - 18 + 2 x 3 = 24 (10 + 18 + 24 + 28 + 30) / 5 = 22
E - 24 + 2 x 2 = 28
A - 28 + 2 x 1 = 30

b) (b) Aplicando la estrategia de prioridad estricta el orden de atención a los procesos y sus tiempos de retorno son:

B - 6
E - 14 Y el tiempo medio de retorno es
A - 24 (6 + 14 + 24 + 26 + 30) / 5 = 20
C - 26
D - 30

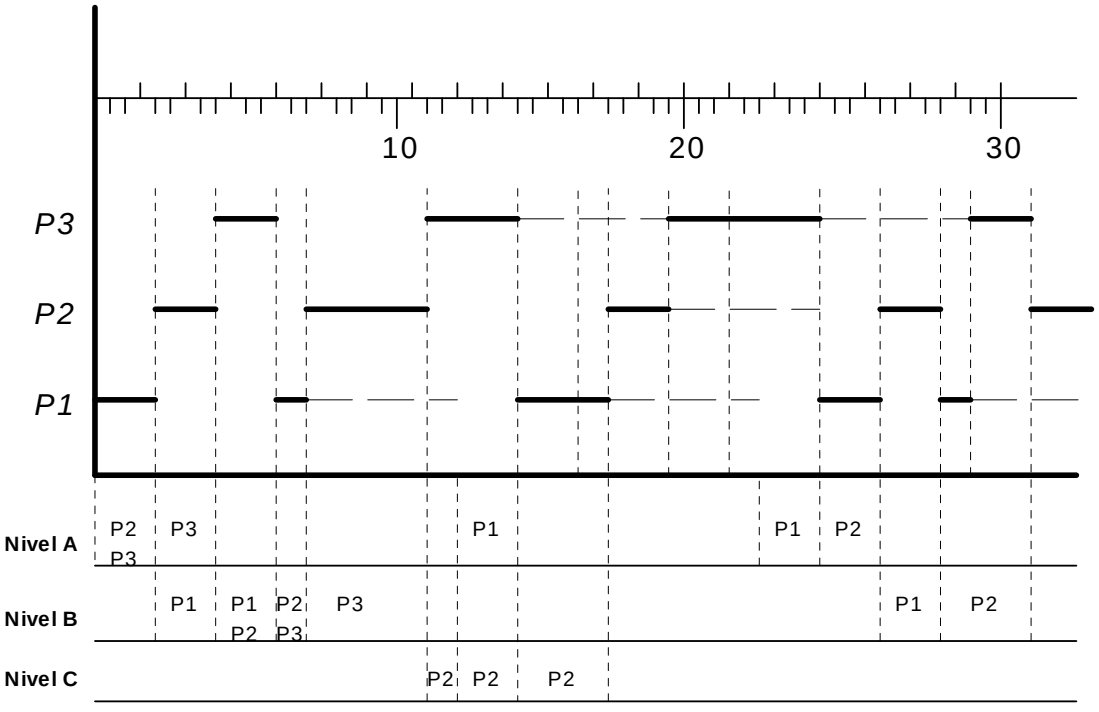
c) (c) Aplicando la estrategia de primero-en-llegar/primero-en-ser-atendido el orden de atención a los procesos y sus tiempos de retorno son:

A - 10
B - 16 Y el tiempo medio de retorno es
C - 18 (10 + 16 + 18 + 22 + 30) / 5 = 19,2
D - 22
E - 30

d) (d) Aplicando la estrategia de primero-el- trabajo-más-corto el orden de atención a los procesos y sus tiempos de retorno son:

C - 2
D - 6 Y el tiempo medio de retorno es
B - 12 (2 + 6 + 12 + 20 + 30) / 5 = 14
E - 20
A - 30

6. Considere un sistema con una política de planificación de procesos …



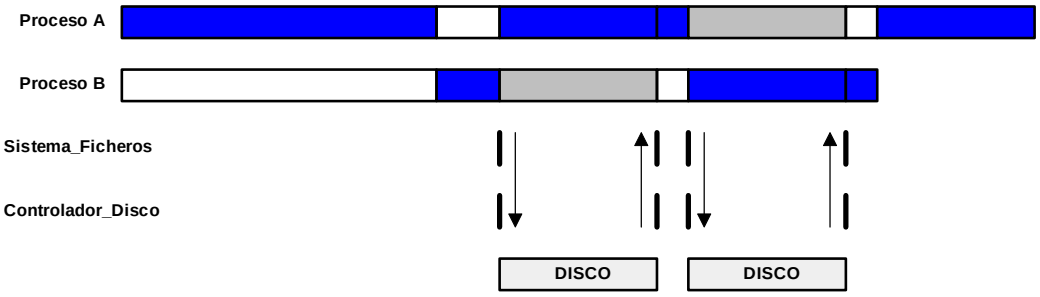
Uso de CPU: 100%
Tiempo de espera de P1: ((12-8) + (10-8) + (12-8))/3 = 3.33
Tiempo de espera de P2: (24-13) = 11
Tiempo de espera de P3: ((19-10) + (10-10)) / 2 = 4.5

7. Considere un sistema con una planificación *MLF* …
8. Repita el ejercicio anterior pero suponiendo que Ti = 2 · q …
9. En un sistema UNIX se quiere programar un proceso que lea de un archivo …

   Ejercicio 2.19 de "Problemas de Sistemas Operativos" [Pérez Costoya et al.]
10. Se tiene un proceso `productor` y otro `consumidor`…

    Ejercicio 2.22 de "Problemas de Sistemas Operativos" [Pérez Costoya et al.]
11. Sea un sistema operativo que sigue el modelo cliente-servidor …

    Ejercicio 3.11



12. Determinar el rendimiento (%uso de CPU y %sobrecarga del S.O.)…