

ADMINISTRACIÓN DE PROCESOS

1. Administración de procesos

NUMERO DE LISTA: 16 2020630160

NOMBRE: GARCIA BOYZO ELIZABETH

GRUPO: 4CV3

TAREA 2

1. ¿QUÉ ES EL PSEUDOPARALELISMO?

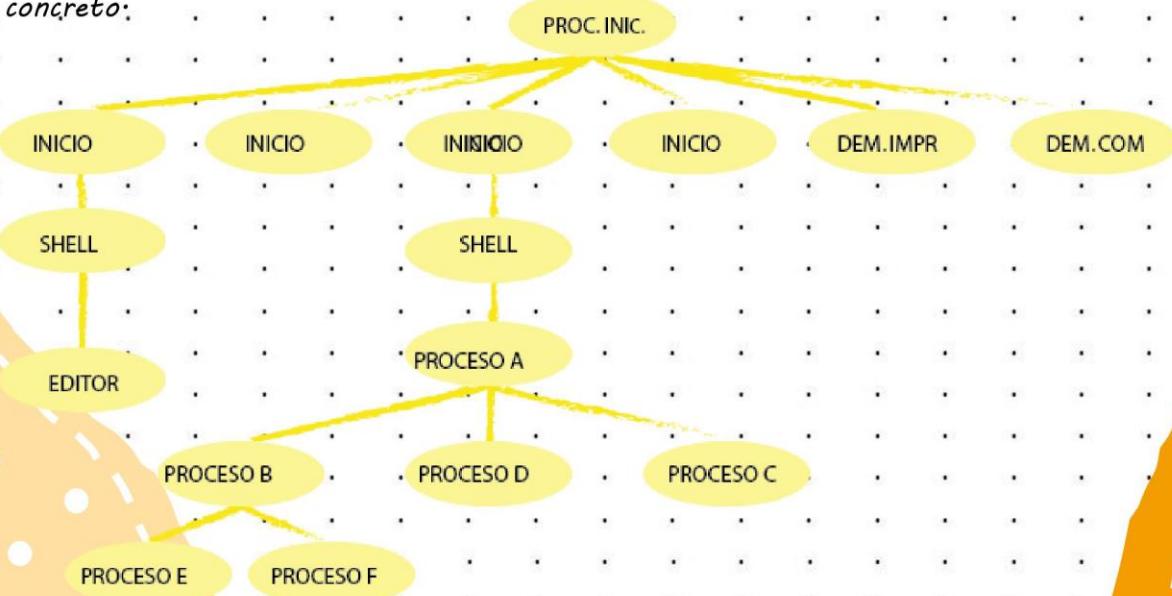
El sistema sólo dispone de una CPU y la concurrencia se consigue mediante períodos de tiempo de ejecución a cada tarea. El control y programación de sistemas multitarea es complicado, almacenado en un dispositivo secundario (por ejemplo un disco).

2. ¿QUÉ ES UN PROCESO?

Un proceso es un programa en ejecución, incluyendo el valor del program counter, los registros y las variables. Conceptualmente, cada proceso tiene un hilo (thread) de ejecución que es visto como un CPU virtual. El recurso procesador es alternado entre los diferentes procesos que existan en el sistema, dando la idea de que ejecutan en paralelo (multiprogramación).

3. DESCRIBA EN QUÉ CONSISTE LA JERARQUÍA DE PROCESOS.

En un sistema lo suficientemente sencillo es posible que todos los procesos que podrían ser necesarios en algún momento puedan estar presentes durante la inicialización del sistema, pero en la mayoría de los sistemas, es necesario una forma de crear y destruir procesos, cuando se requiera durante la operación. Desde el punto de vista macro, los procesos son las actividades claves que se requieren para manjar y dirigir una organización, es necesario mostrar la jerarquía de proceso en la siguiente gráfica. Esta jerarquía muestra cinco niveles: nivel macroproceso, nivel proceso, nivel subproceso, nivel actividades y nivel de tareas específicas a realizar en un proceso concreto:



4. DEFINA QUÉ ES EL BLOQUE DE CONTROL DEL PROCESO O BCP (PROCESS CONTROL BLOCK).

Es un registro especial donde el sistema operativo agrupa toda la información que necesita conocer respecto a un proceso particular. Cada vez que se crea un proceso el sistema operativo crea el BCP correspondiente para que sirva como descripción en tiempo de ejecución durante toda la vida del proceso. Cuando el proceso termina, su BCP es borrado y el registro puede ser utilizado para otros procesos.

5. ¿QUÉ INCLUYE EL BLOQUE DE CONTROL DEL PROCESO?

- Identificador de proceso (Process Identifier -PID-, de sus siglas en inglés).
- Estado del proceso. Por ej: listo, en espera, bloqueado.
- Contador de programa: dirección de la próxima instrucción a ejecutar.
- Valores de registro de CPU. Se utilizan también en el cambio de contexto.
- Espacio de direcciones de memoria.
- Prioridad en caso de utilizarse dicho algoritmo para planificación de CPU.
- Lista de recursos asignados (incluyendo descriptores de archivos y sockets abiertos).
- Estadísticas del proceso.
- Datos del propietario (owner).
- Permisos asignados.
- Señales (Signals) pendientes de ser servidas. (Almacenados en un mapa de bits).

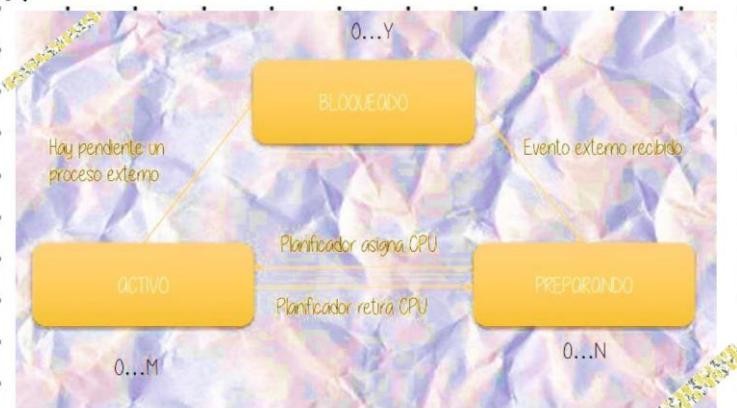
6. DETALLE LAS OPERACIONES QUE SE PUEDEN REALIZAR CON LOS PROCESOS.

- Identificador de proceso (Process Identifier -PID-, de sus siglas en inglés).
- Estado del proceso. Por ej: listo, en espera, bloqueado.
- Contador de programa: dirección de la próxima instrucción a ejecutar.
- Valores de registro de CPU. Se utilizan también en el cambio de contexto.
- Espacio de direcciones de memoria.
- Prioridad en caso de utilizarse dicho algoritmo para planificación de CPU.
- Lista de recursos asignados (incluyendo descriptores de archivos y sockets abiertos).
- Estadísticas del proceso.
- Datos del propietario (owner).
- Permisos asignados.
- Señales (Signals) pendientes de ser servidas. (Almacenados en un mapa de bits).

7. ¿CUÁLES SON LAS OPCIONES DE FINALIZACIÓN DE UN PROCESO?

Terminado: La transición de activo a este estado ocurre cuando el proceso realiza una llamada al sistema solicitando su propia terminación. En estas circunstancias, hay estructuras de datos correspondientes al proceso que no pueden ser liberadas hasta que el proceso padre del que está terminando recoja el código de terminación de este. Hasta que esto ocurra, estas estructuras se mantendrán y el proceso seguirá existiendo en estado terminado.

8. ¿CUÁLES SON LOS TRES ESTADOS COMUNES EN LOS QUE SE PUEDE ENCONTRAR UN PROCESO Y CUÁLES SON SUS TRANSICIONES?



9. DESCRIBA COMO SE REALIZA LA IMPLEMENTACIÓN DE PROCESOS, MEDIANTE UNA TABLA DE PROCESOS Y VECTORES DE INTERRUPCIÓN.

IMPLEMENTACIÓN DE PROCESOS	
FASE DEL PROCESO	DESCRIPCIÓN
En creación:	El núcleo está obteniendo los recursos que necesita el proceso para poder correr, como por ejemplo memoria o disco.
Corriendo (RUN):	El proceso está en posesión del procesador, el que ejecuta sus instrucciones.
Esperando (WAIT):	El proceso espera que se lea un sector del disco, que llegue un mensaje de otro proceso, que transcurra un intervalo de tiempo, que termine otro proceso, etc.
Listo (READY):	El proceso está activo pero no está en posesión del procesador.
Vectores de interrupción:	En muchas arquitecturas de computación típicas, los vectores de interrupción se almacenan en una tabla en una zona de memoria.
Terminado:	El proceso terminó su ejecución, pero sigue existiendo para que otros procesos puedan determinar que terminó.

IRQ	Número de interrupción	Descripción
0	8	Temporizador del sistema (18.2 veces/segundo)
1	9	Teclado
2	0Ah	Controlador de interrupciones programable
3	0Bh	COM2 (puerto serial 2)
4	0Ch	COM1 (puerto serial 1)
5	0Dh	LPT2 (puerto paralelo 2)
6	0Eh	Controlador de disco flexible
7	0Fh	LPT1 (puerto paralelo 1)
8	70h	Reloj CMOS en tiempo real
9	71h	(Se redirige hacia INT 0Ah)
10	72h	(Disponible) tarjeta de sonido
11	73h	(Disponible) tarjeta SCSI
12	74h	Ratón PS/2
13	75h	Coprocessador matemático
14	76h	Controlador de disco duro
15	77h	(Disponible)

10. ¿CUÁL ES LA DIFERENCIA ENTRE UN HILO Y UN PROCESO?

Los hilos, a diferencia de los procesos, no son independientes entre sí. Como todos los hilos pueden acceder a todas las direcciones de la tarea, un hilo puede leer la pila de cualquier otro hilo o escribir sobre ella. Aunque pueda parecer lo contrario la protección no es necesaria ya que el diseño de una tarea con múltiples hilos tiene que ser un usuario único.

II. ¿QUÉ SON LOS HILOS?

Los hilos son un concepto relativamente nuevo de los SO. En este contexto, un proceso recibe el nombre de proceso pesado, mientras que un hilo recibe el nombre de proceso ligero. El término hilo se refiere sintácticamente a hilos de ejecución. El término multihilo hace referencia a la capacidad de un SO para mantener varios hilos de ejecución dentro del mismo proceso.

12. LISTE LOS ESTADOS EN LOS QUE SE PUEDE ENCONTRAR UN HILO.

- Creación: Cuando se crea un proceso se crea un hilo para ese proceso. Luego, este hilo puede crear otros hilos dentro del mismo proceso, proporcionando un puntero de instrucción y los argumentos del nuevo hilo. El hilo tendrá su propio contexto y su propio espacio de la columna, y pasará al final de los Listos.
- Bloqueo: Cuando un hilo necesita esperar por un suceso, se bloquea (salvando sus registros de usuario, contador de programa y punteros de pila). Ahora el procesador podrá pasar a ejecutar otro hilo que esté al principio de los Listos mientras el anterior permanece bloqueado.
- Desbloqueo: Cuando el suceso por el que el hilo se bloqueó se produce, el mismo pasa a la final de los Listos.
- Terminación: Cuando un hilo finaliza se liberan tanto su contexto como sus columnas.

13. ¿POR QUÉ ES NECESARIA LA COMUNICACIÓN ENTRE PROCESOS?

El sistema operativo provee mínimamente dos primitivas, enviar y recibir, normalmente llamadas send y receive. Asimismo, debe implementarse un enlace de comunicación entre los procesos de la comunicación. Este enlace puede ser unidireccional o multidireccional según permita la comunicación en sólo uno o en varios sentidos.

Tipos de comunicación:

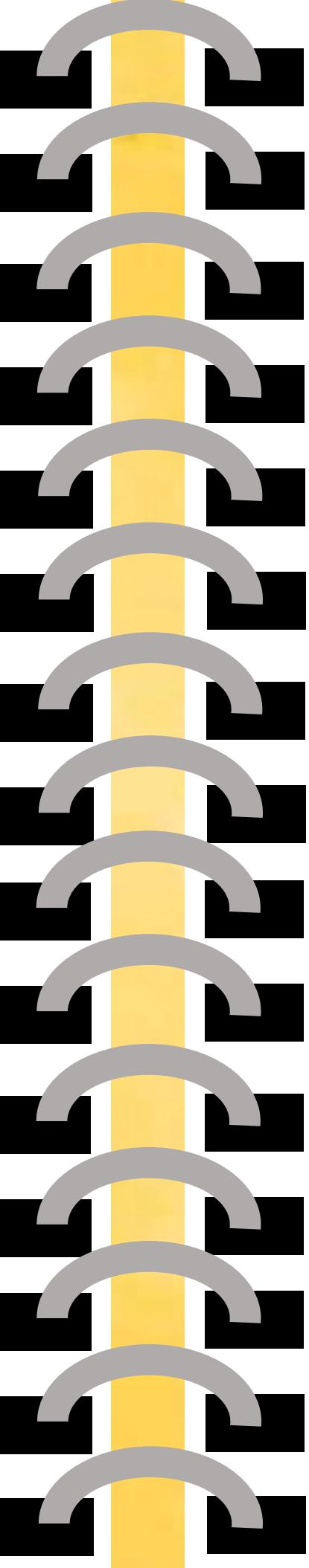
- Síncrona Quien envía permanece bloqueado esperando a que llegue una respuesta del receptor antes de realizar cualquier otro ejercicio.
- Asíncrona Quien envía continúa con su ejecución inmediatamente después de enviar el mensaje al receptor.
- Persistente El receptor no tiene que estar operativo al mismo tiempo que se realiza la comunicación, el mensaje se almacena tanto tiempo como sea necesario para poder ser entregado (Ej.: e-Mail).
- Momentánea (transient) El mensaje se descarta si el receptor no está operativo al tiempo que se realiza la comunicación. Por lo tanto no será entregado.
- Directa Las primitivas enviar y recibir explicitan el nombre del proceso con el que se comunican. Ejemplo: enviar (mensaje, A) envía un mensaje al proceso A. Es decir, se debe especificar cuál va a ser el proceso fuente y cuál va a ser el proceso Destino. Las operaciones básicas Send y Receive se definen de la siguiente manera: Send (P; mensaje); envía un mensaje al proceso P (P es el proceso destino). Receive (Q; mensaje); espera la recepción de un mensaje por parte del proceso Q (Q es el proceso fuente). Nota: Receive puede esperar de un proceso cualquiera, un mensaje, pero el Send sí debe especificar a quién va dirigido y cuál es el mensaje.
- Indirecta La comunicación Indirecta: Es aquella donde la comunicación está basada en una herramienta o instrumento ya que el emisor y el receptor están a distancia.
- Simétrica Todos los procesos pueden enviar o recibir. También llamada bidireccional para el caso de dos procesos.
- Asimétrica Un proceso puede enviar, los demás procesos solo reciben. También llamada unidireccional. Suele usarse para hospedar servidores en Internet.
- Uso de buffers automático El transmisor se bloquea hasta que el receptor recibe el mensaje (capacidad cero).

14. DEFINA LA MEMORIA COMPARTIDA.

La memoria compartida es aquel tipo de memoria que puede ser accedida por múltiples programas, ya sea para comunicarse entre ellos o para evitar copias redundantes. La memoria compartida es un modo eficaz de pasar datos entre aplicaciones.

15. ¿QUÉ ES EL PASO DE MENSAJES?

El modelo de paso de mensajes es el que define los métodos y funciones para poder llevar a cabo el envío de un mensaje de un proceso emisor a un destinatario. Supone un enfoque opuesto al paradigma tradicional en el cual los procesos, funciones y subrutinas sólo podían ser llamados directamente a través de su nombre.



16. DIGA QUE ES UNA TUBERÍA.

Consiste en una cadena de procesos conectados de forma tal que la salida de cada elemento de la cadena es la entrada del próximo. Permiten la comunicación y sincronización entre procesos.

17. ¿CUÁLES SON LAS CARACTERÍSTICAS DE UNA TUBERÍA?

Las tuberías están implementadas en forma muy eficiente en los sistemas operativos multitarea, iniciando todos los procesos al mismo tiempo, y atendiendo automáticamente los requerimientos de lectura de datos para cada proceso cuando los datos son escritos por el proceso anterior. De esta manera el planificador de corto plazo va a dar el uso de la CPU a cada proceso a medida que pueda ejecutarse minimizando los tiempos muertos. Para mejorar el rendimiento, la mayoría de los sistemas operativos implementan las tuberías usando búferes, lo que permite al proceso proveedor generar más datos que lo que el proceso consumidor puede atender inmediatamente.

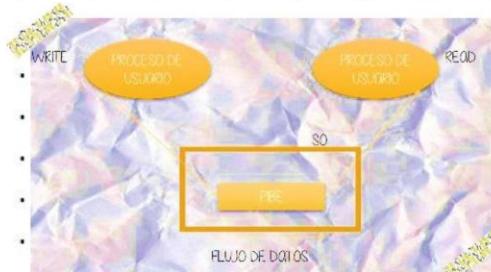
Tubería sin nombre

Las tuberías sin nombre tienen asociado un fichero en memoria principal, por lo tanto, son temporales y se eliminan cuando no están siendo usados ni por productores ni por consumidores. Permiten la comunicación entre el proceso que crea un cauce y procesos hijos tras la creación de la tubería.

Tubería con nombre

Su diferencia respecto a las tuberías sin nombre radica en que el cauce se crea en el sistema de archivos, y por lo tanto no tienen carácter temporal. Se manejan mediante llamadas al sistema (open, close, read y write) como el resto de ficheros del sistema. Permiten la comunicación entre los procesos que usen dicha tubería, aunque no exista una conexión jerárquica entre ellos.

18. MENCIONE UN EJEMPLO DE TUBERÍA.



19. MENCIONE ALGUNAS DE LAS RAZONES PARA SINCRONIZAR PROCESOS/HILOS.

Todos los hilos comparten el mismo espacio de direcciones y otros recursos como pueden ser archivos abiertos. Cualquier modificación de un recurso desde un hilo afecta al entorno del resto de los hilos del mismo proceso. Por lo tanto, es necesario sincronizar la actividad de los distintos hilos para que no interfieran unos con otros o corrompan estructuras de datos. Una ventaja de la programación multihilo es que los programas operan con mayor velocidad en sistemas de computadores con múltiples CPUs (sistemas multiprocesador o a través del grupo de máquinas) ya que los hilos del programa se prestan verdaderamente para la ejecución concurrente. En tal caso el programador necesita ser cuidadoso para evitar condiciones de carrera (problema que sucede cuando diferentes hilos o procesos alteran datos que otros también están usando), y otros comportamientos no intuitivos. Los hilos generalmente requieren reunirse para procesar los datos en el orden correcto. Es posible que los hilos requieran de operaciones átómicas para impedir que los datos comunes sean cambiados o leídos mientras estén siendo modificados, para lo que usualmente se utilizan los semáforos. El descuido de esto puede generar interbloqueo.

20. ¿QUÉ ES UNA CONDICIÓN DE CARRERA?

Ocurre cuando dos o más procesos acceden un recurso compartido sin control, de manera que el resultado combinado de este acceso depende del orden de llegada.

21. DEFINA QUÉ ES UNA SECCIÓN CRÍTICA.

Se denomina región crítica, en programación concurrente de ciencias de la computación, a la porción de código de un programa de ordenador en la que se accede a un recurso compartido que no debe ser accedido por más de un proceso o hilo en ejecución.

22. ¿QUÉ ES LA EXCLUSIÓN MUTUA?

Los algoritmos de exclusión mutua se usan en programación concurrente para evitar que entre más de un proceso a la vez en la sección crítica. La sección crítica es el fragmento de código donde puede modificarse un recurso compartido.

23. ¿CÓMO DEFINE LA ATOMICIDAD?

La atomicidad es la propiedad que asegura que una operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias. Se dice que una operación es atómica cuando es imposible por menos de otra parte de un sistema encontrar pasos intermedios.

24. DESCRIBA UN EJEMPLO DE ABRAZO MORTAL.

Considere un sistema con una impresora y una unidad de disco: Suponga que el proceso A tiene asignada la unidad de disco y que el proceso B tiene asignada la impresora. Ahora, si A pide la impresora y B pide la unidad de disco, ocurre un abrazo mortal.

25. ¿QUÉ ES LA EXCLUSIÓN MUTUAL CON ESPERA ACTIVA?

En un sistema multiprocesador de memoria compartida, se usa la operación indivisible test-and-set sobre una bandera, para esperar hasta que el otro procesador la despeje. La operación test-and-set realiza ambas operaciones sin liberar el bus de memoria a otro procesador. Así, cuando el código deja la sección crítica, se despeja la bandera. Esto se conoce como spin lock o espera activa.

26. ¿QUÉ ES LA EXCLUSIÓN MUTUAL CON ESPERA PASIVA?

Se presentan a continuación varias proposiciones para lograr la exclusión mutua, de manera que mientras un proceso esté ocupado actualizando la memoria compartida en su región crítica, ningún otro proceso puede entrar a su región crítica y ocasionar problemas.

27. MENCIONE LAS CUATRO CONDICIONES PARA EVITAR CONDICIONES DE COMPETENCIA, EMPLEANDO SECCIONES CRÍTICAS

1. Dos procesos no deben encontrarse al mismo tiempo dentro de sus secciones críticas.
2. No se deben hacer hipótesis sobre la velocidad o el número de UCP.
3. Ninguno de los procesos que estén en ejecución fuera de su sección crítica puede bloquear a otros procesos.
4. Ningún proceso debe esperar eternamente para entrar a su sección crítica.

28. ¿QUÉ SON LOS SEMÁFOROS?

Los semáforos son un mecanismo de sincronización de procesos inventados por Edsger Dijkstra en 1965. Los semáforos permiten al programador asistir al planificador del sistema operativo en su toma de decisiones de manera que permiten sincronizar la ejecución de dos o más procesos. A diferencia de los cerros, los semáforos nos ofrecen un mecanismo de espera no ocupada.

29. ¿CÓMO SE RESUELVE EL PROBLEMA DEL PRODUCTOR CONSUMIDOR CON SEMÁFOROS?

1. Dos productores decrementan emptyCount.
2. Un productor determina el siguiente espacio donde insertar el objeto.
3. El segundo productor determina el siguiente espacio para insertar el objeto y resulta ser el mismo del primer productor.

30. ¿QUÉ ES UN MONITOR Y COMO RESUELVE EL PROBLEMA DEL PRODUCTOR CONSUMIDOR?

MONITOR

Un monitor es una estructura del lenguaje cuyas principales características son: Los datos son privados. Ofrecen una serie de métodos públicos para acceder a dichos datos. En cada momento sólo puede haber un proceso activo en algún método del monitor, es decir, ejecutando código de esos métodos públicos del monitor.

PROBLEMA

```
semaphore.fillCount = 0; // items produced
semaphore.emptyCount = BUFFER_SIZE; // remaining space.
procedure producer() {
    while (true) {
        item = producItem();
        down(emptyCount);
        putItemInBuffer(item);
        up(fillCount);
    }
}

procedure consumer() {
    while (true) {
        down(fillCount);
        item = removeItemFromBuffer();
        up(emptyCount);
        consumeItem(item);
    }
}
```

La solución anterior funciona perfectamente cuando hay solo un productor y un consumidor. Cuando múltiples consumidores o productores comparten el mismo espacio de memoria para almacenar el buffer la solución anterior puede llevar a resultados donde dos o más procesos lean o escriban la misma región al mismo tiempo. Para entender como puede ser esto posible imagine como puede ser implementada la función putItemInBuffer(). Podría contener dos acciones, una busca un posible espacio y la otra escribe en él. Si la función puede ejecutarse concurrentemente por múltiples productores entonces el siguiente escenario es posible:

Dos productores decrementan emptyCount
Un productor determina el siguiente espacio donde insertar el objeto.
El segundo productor determina el siguiente espacio para insertar el objeto y resulta ser el mismo del primer productor.
Ambos intentan escribir al mismo tiempo.

31. ¿QUÉ ES LA PLANIFICACIÓN DE PROCESOS?

La planificación de procesos se refiere a cómo determina el sistema operativo al orden en que irá cediendo el uso del procesador a los procesos que lo vayan solicitando, y a las políticas que empleará para que el uso que den a dicho tiempo no sea excesivo respecto al uso esperado del sistema.

32. ¿CUÁLES SON LOS CRITERIOS PARA TENER UN BUEN ALGORITMO DE PLANIFICACIÓN?

- Utilización de la CPU: porcentaje de tiempo que está ocupado el procesador. Debe tender a ser el máximo posible.
- Productividad: trabajos que se completan o finalizan por unidad de tiempo.
- Tiempo de retorno: intervalo de tiempo desde la entrada de un proceso en el sistema hasta su finalización.
- Tiempo de espera: tiempo que un proceso permanece en la cola de preparados.
- Tiempo de respuesta: tiempo que un proceso bloqueado tarda en entrar en la CPU desde que ocurre el evento que lo bloquea.

33. ¿QUÉ ES LA PLANIFICACIÓN APROPIATIVA Y LA NO EXPROPIATIVA?

APROPIATIVA

Permiten que un proceso con mayor prioridad quite la CPU al proceso que está ejecutando antes de su finalización. Los inconvenientes son el coste en pérdidas de tiempo al cambiar de proceso, la coordinación del acceso a datos del núcleo pueda quedar inconsistentes por los cambios de contexto.

NO APROPIATIVA

Una vez que un proceso obtiene la CPU no se le puede quitar hasta que no acabe su ejecución. Esta política genera un problema de acaparamiento injusto de la CPU.

34. DESCRIBA LA PLANIFICACIÓN ROUND ROBIN.

Es un algoritmo de planificación de procesos simple de implementar, dentro de un sistema operativo se asigna a cada proceso una porción de tiempo equitativa y ordenada, tratando a todos los procesos con la misma prioridad. En Sistemas operativos, la planificación Round-robin da un tiempo máximo de uso de CPU a cada proceso, pasado el cual es desalojado y retorna al estado de listo, la lista de procesos se planifica por FIFO, del inglés "First In, First Out" (primero en entrar, primero en salir o primero llegado, primero atendido).

Pasos de ciclos

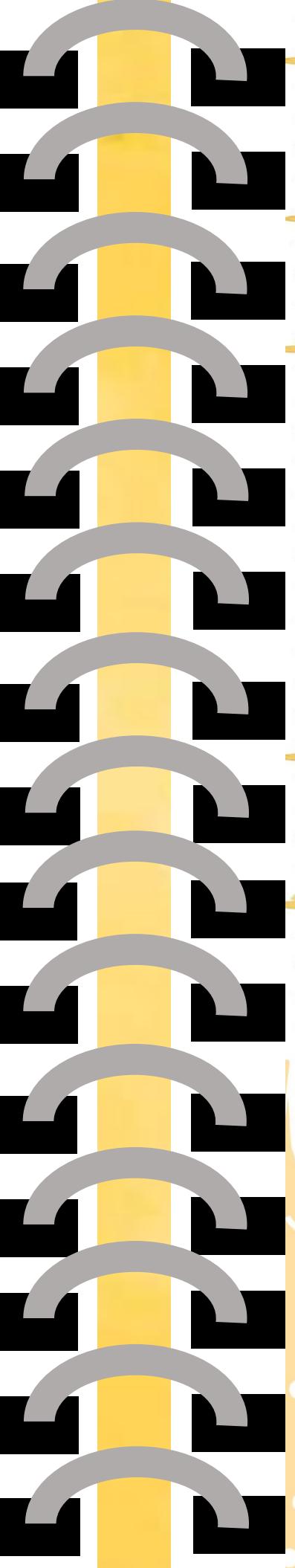
Para averiguar los pasos de ciclos de procesos totales se toman todos los números de procesos y se calculan con los procesos necesarios para la realización de estos...

Suponga que hay tres procesos y se desea averiguar cuánto tarda el proceso A: 3 veces.

proceso B: 4 veces.

proceso C: 5 veces.

$$tp = tProceso + [(tProcesoultimo - tProceso) - (tProcesoultimo - tProcesoproximo)] + tProceso$$



35. ¿QUÉ ES LA PLANIFICACIÓN POR PRIORIDAD?

Consiste en el máximo aprovechamiento del sistema, lo que implica proveer un buen servicio a los procesos existentes en un momento dado. En otras palabras, busca equidad (que todos los procesos sean atendidos).

36. ¿QUÉ ES LA PLANIFICACIÓN POR COLAS MÚLTIPLES?

La planificación mediante colas multinivel es un algoritmo de planificación de procesos en un sistema operativo

37. ¿CÓMO FUNCIONA LA PLANIFICACIÓN POR EL TRABAJO MÁS CORTO?

- Asociar a cada proceso el tiempo de ráfaga de CPU: una vez llega cada proceso se le asigna tiempo de ráfaga de CPU, que son las unidades de tiempo que requieren para que el proceso se ejecute completamente.
- Selecciona el proceso con menor ráfaga de CPU: antes de ejecutar cada proceso el algoritmo prioriza cual es el proceso más corto a ejecutar.
- En caso de empate, aplicar FIFO: si llega a presentarse empate en dos o más procesos de rafaga de CPU, el algoritmo cambia a priorizar primero en entrar primero en salir, algoritmo FIFO.
- Algoritmo no expulsivo: una vez se empieza a ejecutar cada proceso no será interrumpido hasta finalizar.

38. ¿QUÉ ES LA PLANIFICACIÓN GARANTIZADA?

Es un método completamente distinto, que permite hacer promesas reales a los usuarios acerca del rendimiento y después cumplirlas.

39. DESCRIBA LA PLANIFICACIÓN POR LOTERÍA.

La idea básica consiste en dar a los procesos boletos de lotería para los diversos recursos del sistema, como el tiempo de CPU: Cada vez que se hace necesario tomar una decisión de planificación, se escoge al azar un boleto de lotería, y el proceso poseedor de este boleto obtiene el recurso.

Características:

Es de carácter aleatorio:

- Cada vez que aparece un proceso nuevo, se le conceden boletos, con lo que ya tendrá una probabilidad de ganar proporcional al número de boletos recibidos.
- Este esquema de planificación es de resultados comparativamente rápidos en relación a lo pretendido con las reparticiones de boletos, pues tarda solamente hasta el próximo sorteo.
- Se pueden dar más boletos a los procesos más importantes, a fin de aumentar sus posibilidades de ganar.
- En procesos cooperativos pueden intercambiar boletos entre sí como en servicios cliente-servidor donde el cliente puede prestarle temporalmente al servidor sus boletos para una mayor posibilidad de ganar tiempos.
- En este tipo de planificación, la prioridad queda determinada por la cantidad de boletos asignados a un proceso, que influirán estadísticamente de acuerdo a dicha proporción.

40. MENCIONE EN QUÉ CONSISTE LA PLANIFICACIÓN EN TIEMPO REAL.

Consiste en asignar tareas al procesador. La relación biunívoca entre acciones y procesadores es un plan de ejecución (scheduler) y el módulo que hace esto es el planificador (scheduler); para ello utiliza un algoritmo de planificación.

41. ¿CÓMO SE PLANIFICAN LOS PROCESOS EN UNIX?

Cada proceso tiene asignada una prioridad de planificación que cambia con el tiempo. Dicha prioridad le hace pertenecer a una de las múltiples colas de prioridad que maneja el planificador. El planificador siempre selecciona al proceso que encontrándose en el estado preparado en memoria principal para ser ejecutado o en el estado expandido tiene la mayor prioridad. En el caso de los procesos de igual prioridad (se encuentran en la misma cola), lo que hace es ceder el uso de la CPU a uno de ellos durante un cuanto, cuando finaliza dicho cuanto le expropia la CPU y se lo cede a otro proceso. El planificador varía dinámicamente la prioridad de los procesos basándose en su tiempo de uso de la CPU. Si un proceso de mayor prioridad alcanza el estado preparado en memoria principal para ser ejecutado, el planificador expropia el uso de la CPU al proceso actual incluso aunque éste no haya completado su cuanto.

42. ¿CÓMO SE PLANIFICAN LOS HILOS?

El éxito de la planificación de la CPU depende de una propiedad de los procesos:

- La ejecución de un proc. comienza con una ráfaga de CPU, seguida por una ráfaga de E/S, que es seguida por otra ráfaga de CPU, luego otra ráfaga de E/S; y así sucesivamente.
- Finalmente, la ráfaga final de CPU concluye con una solicitud al sistema para terminar la ejecución.
- Un programa ligado por E/S tiene generalmente muchas ráfagas de CPU cortas.
- Un programa ligado por CPU tiene generalmente pocas ráfagas de CPU largas.
- Esta distribución es importante al implementar un algoritmo de planificación de CPU.
- Las características que se utilizan en la comparación pueden marcar una diferencia sustancial en qué algoritmo se considera mejor. Los criterios incluyen lo siguiente:
- Utilización de la CPU; mantener la CPU lo más ocupada posible. Conceptualmente puede variar de 0 a 100%, pero en un sistema real debe oscilar entre un 40% y un 90%.
- Throughput (Rendimiento): número de proc. que se completan por unidad de tiempo.
- Turnaround time (Tiempo de retorno o tiempo de ejecución): corresponde al tiempo transcurrido desde el lanzamiento de un proc. hasta su finalización. Es la suma de los períodos pasados esperando en la cola de listos, ejecutándose en la CPU y haciendo E/S.
- Waiting time (Tiempo de espera): suma de los períodos invertidos en esperar en la cola de proc. listos.
- Response time (Tiempo de respuesta): es el tiempo que el proc. tarda en empezar a responder (no el tiempo que tarda en enviar a la salida toda la información de respuesta).
- Generalmente, el tiempo de respuesta está limitado por la velocidad del dispositivo de salida.

43. ¿CUÁLES SON LAS POLÍTICAS DE DESALOJO DE PROCESO/HILOS EN UN SISTEMA OPERATIVO?

Permiten forzar la expulsión del ser interrumpido y pasado al estado de listos por el sistema operativo.

44. ¿QUÉ ALGORITMOS SE USAN PARA PLANIFICAR HILOS?

- Planificación tipo round robin.
- Planificación por prioridad.
- Planificación colas múltiples.
- Planificación primero el más corto.
- Planificación de servicio por orden de llegada

REFERENCIAS

- Glez . del . Alba, . Angel: . Teoría. . de . los . Sistemas . Operativos. . s/f .
ieru.org/org/tmp/informatica/ssoo/Apuntes/Cap5sol_a.doc
- Rámirez, Israel J: Los Sistemas Operativos. ULA-FACES. Mérida. s/f.
<http://isis.faces.ulb.ve/COMPUTACIÓN/Israel/SistemaPDF>
- Tanenbaum, Andrew: Sistemas Operativos Modernos. Ed Pearson-Prentice Hall: México, 3ra Edición. 2009..
- Thomas Anderson, Michael Dahlin, *Operating Systems: Principles and Practice* (2.ª ed.), Recursive Books (2014). [DAH]
- Remzi H. Arpaci-Dusseau, Andrea C. Arpaci-Dusseau, *Operating Systems: Three Easy Pieces* (y0.91), Arpaci-Dusseau Books (2015). [ARP]
- Randal E. Bryant, David R. O'Hallaron, *Computer Systems: A Programmer's Perspective* (2.ª ed.), Prentice Hall (2011). [BRY2]
- Michael Kerrisk, *The Linux Programming Interface: A Linux and UNIX System Programming Handbook* (1.ª ed.), No Starch Press. (2010). [KERR]
- Linux Internals
Moshe Bar McGraw-Hill Osborné Media (2000)
- Inside Microsoft Windows 2000 (Microsoft Programming Series)
David A. Solomon, Mark Russinovich Microsoft Press (2000)
- Understanding the Linux Kernel.
Daniel P. Bovet & Marco Cesati O'Reilly (3rd Edition)
- Essential Linux Device Drivers
Sreekrishnan Venkateswaran Prentice Hall
- Sistemas Operativos. Diseño e implementación.
A.S. Tanenbaum, A.S. Woodhull Prentice Hall. 3ª ed. 2006 (ingles), 2ª ed. 1997 (Español)
- Sistemas Operativos. Aspectos interhos y principios de diseño
William Stallings Pearson educación. 5ª edición. 2005.
- Fundamentos de sistemas operativos.
Silberschatz, P. Galvin, G. Gagne McGraw-Hill/Interamericana de España S.A. 7ª.edición. 2006.
- Problemas resueltos de programación en C
F. García, J. Carretero, A. Calderón, J. Fernández, J. M. Pérez Thomson, 2003. ISBN: 84-9732-102-2.