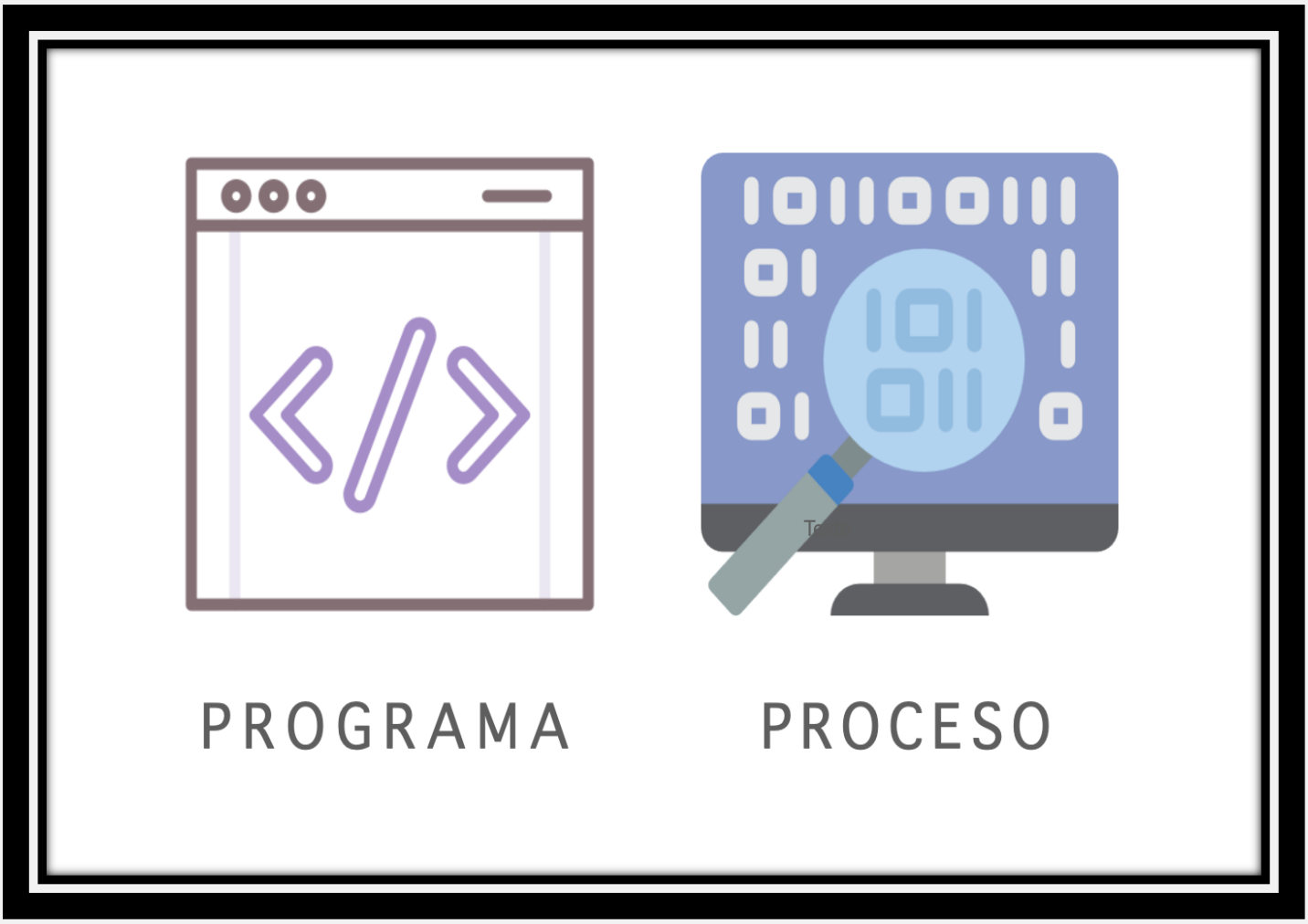


Sistemas Operativos

Tarea no. 2

Administración de procesos

Alumno:

Angelo Mihaelle Ojeda Gómez

Numero de lista:
23

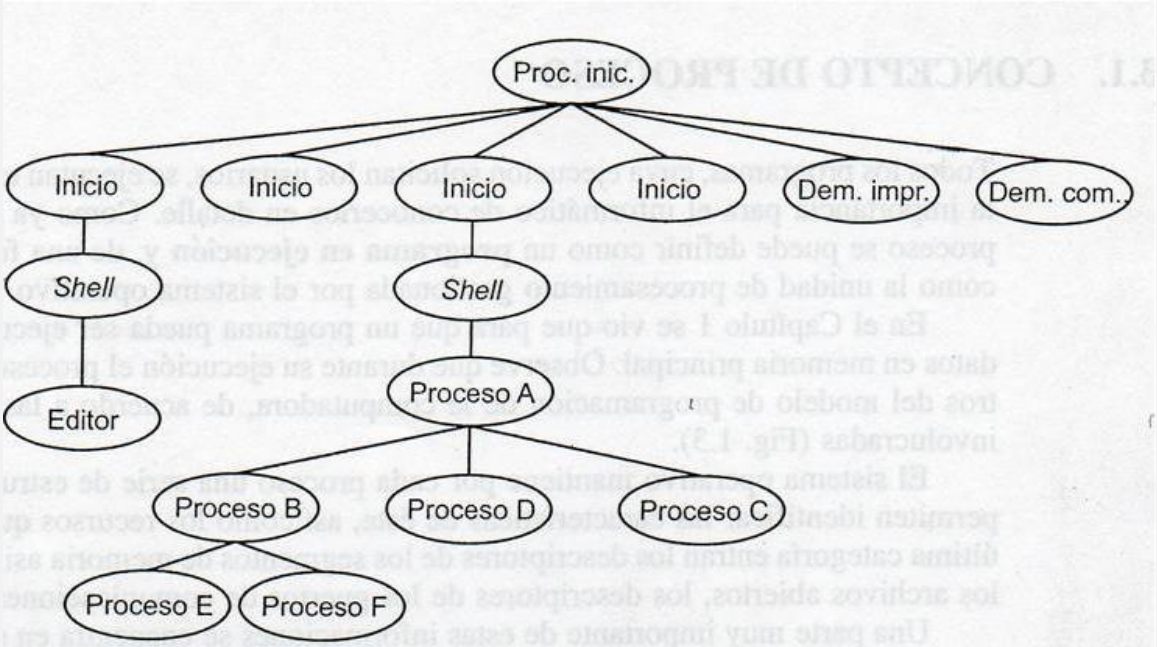
Grupo:

4CV3

(Las respuestas aparecerán de la siguiente forma: "Respuesta"*; con texto en negro.)*

1. *¿Qué es el seudoparalelismo?*
   - En un instante dado, la CPU está ejecutando un solo programa, en el curso de un segundo puede trabajar con varios programas dando a los usuarios la ilusión de paralelismo. A veces se usa el término seudoparalelismo para referirse a esta rápida conmutación de la CPU entre programas, para distinguirla del verdadero paralelismo de hardware de los sistemas multiprocesador (que tienen dos o más CPU que comparten la misma memoria física).
2. *¿Qué es un proceso?*
   - Un proceso es el contexto básico en el que se da servicio a todas las actividades solicitadas por el usuario dentro del sistema operativo. Un proceso no es más que un programa en ejecución, e incluye los valores actuales del contador de programa, los registros y las variables.
     i. Conceptualmente, cada uno de estos procesos tiene su propia CPU virtual. Desde luego, en la realidad la verdadera CPU conmuta de un proceso a otro, pero para entender el sistema es mucho más fácil pensar en una colección de procesos que se ejecutan en (seudo) paralelo que tratar de seguir la pista a la forma en que la CPU conmuta de un programa a otro.

3. *Describa en que consiste la jerarquía de procesos.*
   - Es posible que todos los procesos de un sistema que puedan ser necesarios en algún momento podrían estar presentes durante la inicialización del sistema, pero en la mayoría de los sistemas, es necesario una forma de crear y destruir procesos, cundo se requiera durante la operación. Desde el punto de vista macro, los procesos son las actividades claves que se requieren para manejar y dirigir una organización, es necesario mostrar la jerarquía de proceso en la siguiente grafica. Esta jerarquía muestra cinco niveles:
     i. nivel macroproceso
     ii. nivel proceso
     iii. nivel subproceso,
     iv. nivel actividades
     v. nivel de tareas específicas a realizar en un proceso concreto.
   - La grafica en cuestión seria la siguiente:



4. *Defina qué es el bloque de control del proceso o BCP (Process Control Block).*
    - El bloque de control del proceso (BCP) 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.
        i. Cuando el proceso termina, su BCP es borrado y el registro puede ser utilizado para otros procesos. Un proceso resulta conocido para el sistema operativo y por tanto elegible para competir por los recursos del sistema sólo cuando existe un BCP activo asociado a él. El bloque de control de proceso es una estructura de datos con campos para registrar los diferentes aspectos de la ejecución del proceso y de la utilización de recursos.

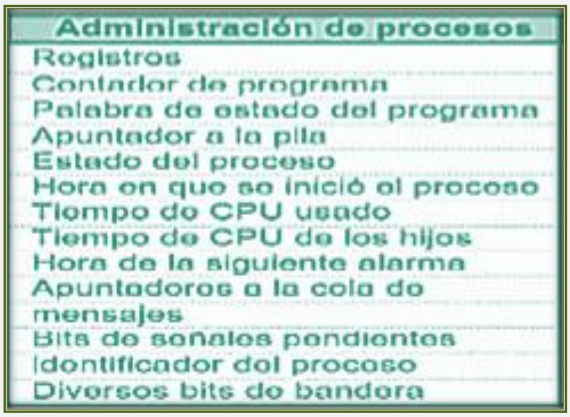
5. *¿Qué incluye el bloque de control del proceso?*
    - La información almacenada en un BCP incluye típicamente algunos o todos los campos siguientes:
        - Identificador de proceso (Process Identificator -PID-, de sus siglas en inglés).
        - Estado del proceso. Por ej: listo, en espera, bloqueado.
        - 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.*
    - Crear: El trabajo para el sistema operativo consiste en darle una entrada en el BCO y pasarlo a la cola de preparados. Hay varias formas de crear procesos:
        i. inicialización del sistema: Cuando se inicia el sistema se crean varios procesos que interactúan o no con el usuario
        ii. Ejecución de una llamada al sistema para crear procesos por parte de una ejecución
        iii. Solicitud de un usuario para crear un proceso.
        iv. Inicio de un trabajo por lotes
    - Destruir: eliminar la entrada en la cola de PCB. Puede haber problemas en la gestión de las propiedades heredadas del proceso padre o, si tiene procesos hijo, tener que esperar a que finalicen estos o los finaliza forzosamente.
        i. Terminación normal. Es la forma mas normal (exit en UNIX)
        ii. Terminación por error. Por ejemplo gcc uno.c y el fichero uno.c no existe.
        iii. Error fatal. Acceso a una posición no permitida, división por cero … etc.
        iv. Terminado por otro proceso. En UNIX es KILL.
    - Dormir o bloquear la ejecución de un proceso.
    - Cambiar la prioridad del proceso.
    - Dormir un proceso un tiempo.
    - Despertar un proceso. Una forma de desbloquear un proceso de forma artificial. Se suele emplear para procesos dormidos artificialmente.
    - Suspender un proceso. Suele hacerse en situaciones de sobrecarga del S.O.
    - Reanudar un proceso. Activar un proceso suspendido.

7. *¿Cuáles son las opciones de finalización de un proceso?*
    - Terminación de un proceso: Un proceso termina cuando ejecuta su última instrucción y pide al sistema operativo que lo elimine. En este momento, el proceso puede devolver un valor de estado a su proceso padre.
    - El sistema operativo libera la asignación de todos los recursos del proceso, incluyendo las memorias física y virtual, los archivos abiertos y los búferes de E\S
    - Razones para la terminación de un proceso:
        i. Normal: El proceso ejecuta llamada al servicio del SO que notifica su terminación normal
        ii. Por tiempo excedido: Una tarea tiene asignada la CPU ejecuta el tiempo máximo y le es requisada la CPU y pasa a la cola para competir por la nueva asignación.
        iii. Violación de límites: Proceso trata de acceder a una posición de memoria que no le está permitida acceder.
        iv. No memoria disponible: El proceso necesita más memoria de la que el sistema puede proporcionar.
        v. Error de protección: El proceso intenta utilizar un recurso o archivo que no le está permitido utilizar, o trata de utilizarlo de forma incorrecta.
        vi. Error aritmético: Si el proceso intenta hacer un cálculo prohibido, como la división por cero, o trata de acceder a un número mayor del que el hardware acepta.
        vii. Tiempo máximo de espera de recurso: El proceso ha esperado más allá del tiempo máximo especificado para que se produzca cierto suceso.
        viii. Fallo de dispositivo de E/S: Se produce un error en una operación de E/S
        ix. Instrucción no valida: El proceso intenta ejecutar una instrucción inexistente (a menudo como resultado de un salto para ejecutar datos en la zona de datos)
        x. Intento de acceso a una instrucción privilegiada: El proceso intenta utilizar una instrucción reservada para el SO.
        xi. Finalización del padre: Cuando un proceso padre finaliza, el SO puede diseñarse para terminar automáticamente con todos sus descendientes.
        xii. Mal uso de los datos: Un elemento de dato, no está inicializado o es de un tipo equivocado.
        xiii. Intervención del operador o del SO: Por alguna razón el operador o el SO termina con un proceso (ej.: Interbloqueo).
        xiv. Solicitud del padre: Un proceso padre tiene normalmente autoridad para terminar con cualquiera de sus hijos.

8. *¿Cuáles son los tres estados comunes en los que se puede encontrar un proceso y cuáles son sus transiciones?*
    - Ejecutándose
    - Listo
    - Bloqueado
        i. Transiciones entre Estados:
            1. Un proceso se bloquea para aceptar entradas (Ejecutandose -> Bloqueado)
            2. El planificador elige otro proceso (Ejecutandose -> Listo)
            3. El planificador elige este proceso (Listo -> Ejecutándose)
            4. Hay entradas disponibles (Bloqueado -> Listo)

9. *Describa como se realiza la implementación de procesos, mediante una tabla de procesos y vectores de interrupción.*
    - Para implementar el modelo de procesos, el sistema operativo mantiene una tabla llamada tabla de procesos, con una entrada por cada proceso. Esta contiene información acerca del estado del proceso, su contador de programa, el apuntador de pila, el reparto de memoria, la situación de sus archivos abiertos, su información de contabilidad (y planificación) y todos los demás aspectos de un proceso que se deben de guardar cuando este se conmuta del estado ejecutándose al estado listo, fin de poder reiniciarlo después como si nunca se hubiera detenido.



10. *¿Cuál es la diferencia entre un hilo y un proceso?*
    - Los hilos se distinguen de los tradicionales procesos en que los procesos son, generalmente, independientes, llevan bastante información de estados, e interactúan solo a través de mecanismos de comunicación dados por el sistema. Por otra parte, muchos hilos generalmente comparten otros recursos de forma directa. En muchos de los sistemas operativos que dan facilidades a los hilos, es más rápido cambiar de un hilo a otro dentro del mismo proceso, que cambiar de un proceso a otro. Este fenómeno se debe a que los hilos comparten datos y espacios de direcciones, mientras que los procesos, al ser independientes, no lo hacen. Al cambiar de un proceso a otro el sistema operativo lo que se conoce como overhead, en este caso pasar del estado de ejecución al estado de espera, o bloqueo, y colocar el nuevo proceso en ejecución. En los hilos, como pertenecen a un mismo proceso, al realizar un cambio de hilo el tiempo perdido es casi inapreciable.

11. *¿Qué son los hilos?*
    - En sistemas operativos, un hilo o hebra (del inglés thread), proceso ligero o subproceso es una secuencia de tareas encadenadas muy pequeña que puede ser ejecutada por un sistema operativo.
    - Un hilo es una unidad básica de utilización de CPU, la cual contiene un id de hilo, su propio programa counter, un conjunto de registros, y una pila; que se representa a nivel del sistema operativo con una estructura llamada TCB (thread control block o Bloque de control de hilos). Los hilos comparten con otros hilos que pertenecen al mismo proceso la sección de código, la sección de datos, entre otras cosas. Si un proceso tiene múltiples hilos, puede realizar más de una tarea a la vez.

12. *Liste los estados en los que se puede encontrar un hilo.*
    - Los principales estados de los hilos son:
        i. Ejecución
        ii. Listo
        iii. Bloqueado

13. *¿Por qué es necesaria la comunicación entre procesos?*
    - Los procesos con frecuencia necesitan comunicarse con otro proceso, Por ejemplo, en un conducto de shell, la salida del primer proceso debe de pasarse al segundo, y así sucesivamente. Por tanto, es necesaria la comunicación entre procesos, de forma estructurada y sin hacer uso de interrupciones, para que todos estos procesos puedan funcionar de una forma coordinada.

14. *Defina la memoria compartida.*
    - La memoria compartida significa que dos o mas procesos comparten la misma área de memoria. Pueden acceder al contenido de esta área. 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. Dependiendo del contexto, los programas pueden ejecutarse en un mismo procesador o en procesadores separados. La memoria usada entre dos hilos de ejecución dentro de un mismo programa se conoce también como memoria compartida
15. *¿Qué es el paso de mensajes?*
    - En ciencias de la computación, el paso de mensajes es un paradigma de programación ampliamente usado en el software moderno. Sus aplicaciones cubren un amplio campo, y puede usarse desde para garantizar que los diferentes objetos que constituyen un programa informático puedan trabajar de forma coherente entre ellos hasta para permitir que una tarea pueda ejecutarse de forma sincronizada entre varios ordenadores.
    - La transferencia de mensajes puede tener muchas formas:
        i. Una de estas formas es asignar a cada proceso una dirección única y hacer que los mensajes se dirijan a cada uno de los procesos
        ii. Otro metodo consiste en inventar una nueva estructura de datos llamada buzón, este, es un luchar donde se almacena temporalmente una cierta cantidad de mensajes predeterminada al crear el buzón, cuando este esta lleno y un proceso intenta transmitir queda suspendido este, se retira un mensaje y se deja espacio para el nuevo
16. *Diga que es una tubería.*
    - La tubería es la forma mas antigua de comunicación entre procesos, y es el mecanismo más básico de IPC. Llamamos tubería al flujo de datos que se conecta de un proceso al otro.

17. *¿Cuáles son las características de una tunería?*
    - La canalización proporciona servicios de transmisión
    - El ciclo de vida sigue el proceso
    - El kernel sincronizará y excluirá mutuamente su canalización
    - La canalización es semidúplex. Cuando dos partes necesitan comunicarse, se establecen 2 canalizaciones
18. *Mencione un ejemplo de tubería.*
    - Usando UNIX shell una tubería se especifica con "|" entre secuencias de dos secuencias de comandos, la salida o resultado del primer comando es usado como entrada del segundo, por ejemplo:

```
1.  $ ls -l | more
```

19. *Mencione algunas de las razones para sincronizar procesos/hilos.*
    - Las variables globales compartidas entre varios hilos y/o procesos:
        i. Si no se comparten todos los recursos, se pierde la intención original del proceso y el diseño del hilo: Compartir recursos mejorar la utilización de estos
    - El orden de ejecución relativo entre procesos/ hilos:
        i. El orden de ejecución relativo entre subprocesos debe determinarse cuando sea necesario

20. *¿Qué es una condición de carrera?*
    - Ocurre cuando dos o mas procesos acceden a un recurso compartido sin control, lo que, combinado con el resultado de este acceso depende del orden de llegada.
    - En algunos sistemas operativos, los procesos que están colaborando podrían compartir cierto almacenamiento común en el que ambos pueden leer y escribir. El almacenamiento compartido puede estar en la memoria principal, o puede tratarse de un archivo compartido; la ubicación de la memoria compartida con altera la naturaleza de la comunicación ni los problemas surgen.

21. *Defina qué es una sección crítica.*
    - Es la porción de Código o programa, en la que se accede a un recurso compartido que no debe de ser accedido por mas de un proceso o hilo de ejecución, por loque se necesita una exclusión mutua en este archivo en cuanto a la apertura de este por parte de un proceso o hilo
22. *¿Qué es la exclusión mutua?*
    - Es la forma de asegurar que si un proceso esta usando una variable o archivo compartido, los procesos quedaran excluidos de hacer lo mismo.
    - Es la actividad que realiza un sistema operativo para evitar que dos o más procesos ingresen al mismo tiempo a un área de datos compartidos, o que se acceda a ellos mas de una vez al mismo tiempo cuando esto no está permitido

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 sin completar. Se dice que una operación o procesos es atómico cuando es imposible por lo mendos que otra parte de un sistema encuentre los pasos intermedios

24. *Describa un ejemplo de abrazo mortal.*
    - El abrazo mortal es el bloqueo permanente de un conjunto de procesos o hilos de ejecución en un sistema concurrente que compiten por recursos del sistema o bien se comunican entre ellos.1 A diferencia de otros problemas de concurrencia de procesos, no existe una solución general para los interbloqueos. Por ejemplo:
        i. Dos procesos compiten por dos recursos que necesitan para funcionar, que sólo pueden ser utilizados por un proceso a la vez. El primer proceso obtiene el permiso de utilizar uno de los recursos (adquiere el lock sobre ese recurso). El segundo proceso toma el lock del otro recurso, y luego intenta utilizar el recurso ya utilizado por el primer proceso, por lo tanto queda en espera

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 despeja. La operación antes mencionada realiza ambas operaciones sin liberal el bus de memoria a otro procesador. Así, cuando el código de la sección critica se despeja la bandera.
26. *¿Qué es la exclusión mutual con espera pasiva?*
    - En un sistema multiprocesador de memoria compartida mientras un proceso esta ocupado, actualizando la memoria compartida en su región critica otro proceso puede entrar a su región critica y ocasionar problemas, por lo que, una técnica donde un proceso repetidamente verifica una condición, tal como esperar una entrada de teclado o si el ingreso a una sección crítica está habilitado, a lo que se le llama espera pasiva
27. *Mencione las cuatro condiciones para evitar condiciones de competencia, empleando secciones críticas.*
    - Dos procesos nunca pueden estar simultáneamente dentro de regiones críticas,
    - No puede suponerse nada acerca de las velocidades o el número de los CPU
    - Ningún proceso que se ejecute fuera de su región critica puede bloquear a otros procesos.
    - Ningún proceso tendrá que esperar indefinidamente para entrar en su región critica.
28. *¿Qué son los semáforos?*
    - Los semáforos son una herramienta de sincronización que ofrece una solución al problema de la sección crítica (porción de código de un programa de computador en la cual se accede a un recurso compartido que no debe ser accedido por más de un proceso o hilo en ejecución). Un semáforo provee una simple pero útil abstracción para controlar el acceso de múltiples procesos a un recurso común en programación paralela, o entornos multiusuarios. El concepto de semáforo fue inventando por el holandés Esdger W. Dijkstra.

29. ¿Cómo se resuelve el problema del productor consumidor con semáforos?

- El uso de semáforos resuelve el problema de las llamadas perdidas. En la solución que se muestra a continuación se usan dos semáforos *fillCount* y *emptyCount*. El primero es el número de objetos que existen en el buffer mientras que *emptyCount* es el número de espacios disponibles donde se puede insertar objetos. *fillCount* es incrementado y *emptyCount* es decrementado cuando se inserta un objeto en el buffer. Si el productor intenta decrementar *emptyCount* cuando su valor es cero, el productor es puesto a dormir. La próxima vez que se consuma un objeto el productor despierta. El consumidor funciona análogamente.

```
1. semaphore fillCount = 0; // items produced
2. semaphore emptyCount = BUFFER_SIZE; // remaining space
3.
4. procedure producer() {
5.     while (true) {
6.         item = produceItem();
7.         down(emptyCount);
8.             putItemIntoBuffer(item);
9.         up(fillCount);
10.     }
11. }
12.
13. procedure consumer() {
14.     while (true) {
15.         down(fillCount);
16.             item = removeItemFromBuffer();
17.         up(emptyCount);
18.         consumeItem(item);
19. }
```

30. ¿Qué es un monitor y como resuelve el problema del productor consumidor?

- El siguiente pseudocódigo muestra una solución al problema usando monitores. Como la exclusión mutua está implícita en los monitores no es necesario un esfuerzo extra para proteger la región crítica.

```
1.  monitor ProducerConsumer {
2.      int itemCount
3.      condition full;
4.      condition empty;
5.
6.      procedure add(item) {
7.          while (itemCount == BUFFER_SIZE) {
8.              wait(full);
9.          }
10.
11.         putItemIntoBuffer(item);
12.         itemCount = itemCount + 1;
13.
14.         if (itemCount == 1) {
15.             notify(empty);
16.         }
17.     }
18.     procedure remove() {
19.         while (itemCount == 0) {
20.             wait(empty);
21.         }
22.
23.         item = removeItemFromBuffer();
24.         itemCount = itemCount - 1;
```

```
25.
26.         if (itemCount == BUFFER_SIZE - 1) {
27.             notify(full);
28.         }
29.
30.         return item;
31.     }
32. }
33.
34. procedure producer() {
35.     while (true) {
36.         item = produceItem()
37.         ProducerConsumer.add(item)
38.     }
39. }
40.
41. procedure consumer() {
42.     while (true) {
43.         item = ProducerConsumer.remove()
44.         consumeItem(item)
45.     }
46. }
```

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. Conjunto de políticas y mecanismos incorporados al sistema operativo, a través de un módulo denominado planificador, que debe decidir cuál de los procesos en condiciones de ser ejecutado conviene ser despachado primero y qué orden de ejecución debe seguirse. Esto debe realizarse sin perder de vista su principal objetivo que consiste en el máximo aprovechamiento del sistema, lo que implica proveer un buen servicio a los procesos existentes en un momento dado.
32. *¿Cuáles son los criterios para tener un buen algoritmo de planificación?*
    - Equitatividad: Asegurarse de que cada proceso reciba una parte justa del tiempo de CPU
    - Eficiencia: Mantener la CPU ocupada todo el tiempo.
    - Tiempo de respuesta: Minimizar el tiempo de respuesta para los usuarios.
    - Retorno: Minimizar el tiempo que los usuarios por lotes tienen que esperar sus salidas.
    - Volumen de producción: Maximizar el número de trabajos procesados por hora.

33. *¿Qué es la planificación expropiativa y la no expropiativa?*
    - Sistemas operativos con planificadores no expropiativos (nonpreemptive) son los que asignan el recurso procesador a un proceso y hasta que este no lo libere, ya sea porque finaliza su ejecución o se bloquea, no se vuelve a ejecutar el planificador.
    - Sistemas operativos con planificadores expropiativos (preemptive) son que los que pueden expropiar el recurso procesador a un proceso cuando otro proceso entra en estado pronto (ya sea porque es nuevo o porque se desbloqueó) o porque se le impone un límite de tiempo para ejecutar.
34. *Describa la planificación Round Robin.*
    - A cada proceso se le asigna un intervalo de tiempo, llamado cuanto, durante el cual se permite ejecutarse. Si el proceso todavía se está ejecutando al expirar su a cuanto, el sistema operativo se apropia de la CPU y se la da a otros procesos. Si el proceso se bloquea o termina antes de expirar el cuanto, la comunicación d CPU naturalmente se efectúa cuando el proceso se bloquee.
35. *¿Qué es la planificación por prioridad?*
    - A cada proceso se le asigna un número entero que representa su prioridad.
    - El planificador asigna el procesador al proceso con la más alta prioridad. Se utiliza en general un esquema expropiativo ya que si un proceso con mayor prioridad que el que está ejecutando arriba a la lista de procesos listos (ready queue), será asignado al procesador.

36. ¿Qué es la planificación por colas múltiples?

- Si los procesos se pueden clasificar según sus cualidades, es posible dividir la lista de procesos listos (ready queue) en varias colas (una para cada clasificación). Los procesos son asignados permanentemente a una de las colas. Cada cola tendrá su propio algoritmo de planificación propio. Además, se debe tener una estrategia de planificación entre las diferentes colas. Por ejemplo, una cola tendrá prioridad sobre otra.

37. ¿Cómo funciona la planificación por el trabajo más corto?

- El algoritmo asocia a los procesos el largo de su próximo CPU-burst. Cuando el procesador queda disponible se le asigna al proceso que tenga el menor CPU-burst. Si dos procesos tienen el mismo CPU-burst se desempata de alguna forma. Su funcionamiento depende de conocer los tiempos de ejecución lo cual en la mayoría de los casos no sucede.

38. ¿Qué es la planificación garantizada?

- Consiste en hacer promesas reales al usuario en cuanto al rendimiento y después de cumplirlas. Se establecen compromisos de desempeño con el proceso del usuario. tiene como objetivo dividir el tiempo del CPU entre los diferentes procesos, de esta forma los organiza y se compromete (promete) a procesarlos conforme al tiempo asignado, al cumplir lo prometido con respecto al tiempo es que garantiza la operación adecuada del proceso. El sistema, maneja un registro del tiempo del CPU, que cada proceso ha tenido desde su llegada al sistema. Con los datos anteriores el registro de proceso en curso de ejecución, el sistema calcula y determina cual de estos esta más alejado por defecto de la relación.

39. Describa la planificación por lotería.

- Consiste en dar a los procesos “Boletos de lotería” para los diversos recursos del sistema, como el tiempo del 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. Es de carácter aleatorio. Cada vez que aparezca 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 con 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.

40. Mencione en qué consiste la planificación en tiempo real.

- Un proceso de tiempo real es aquel cuya actividad tiene un plazo de finalización. Un sistema de tiempo real es un en el que el tiempo desempeña un papel esencial, por lo regular, uno o más dispositivos físicos externos a la computadora generan estímulos, u la computadora debe de reaccionar a ello dentro de un plazo fijo.

41. ¿Cómo se planifican los procesos en UNIX?

- El núcleo tradicional de UNIX es estrictamente no expropiable. Es decir, si el proceso actual se encuentra en modo núcleo (debido a una llamada al sistema o a una interrupción), no puede ser forzado a ceder la CPU a un proceso de mayor prioridad. Dicho proceso cederá voluntariamente la CPU cuando entre en el estado dormido. En caso contrario sólo se le podrá expropiar la CPU cuando retorne a modo usuario.

42. ¿Cómo se planifican los hilos?

- La planificación de ejecución de hilos se basa en el modelo de prioridades y no utiliza el modelo de segmentación por segmentos de tiempo. Un hilo continuará ejecutándose en la CPU hasta pasar a un estado que no le permita seguir en ejecución. Si se quiere alternancia entre hilos, se debe asegurar que el thread permite la ejecución de otros hilos. Por ejemplo, usando sleep().

43. ¿Cuáles son las políticas de desalojo de proceso/hilos en un sistema operativo?

- Este segundo tipo de política permite que el SO le retire el procesador al proceso que lo tenga. De tal forma, a los dos momentos anteriores para planificar un nuevo proceso (terminó, tuvo que hacer una entrada/salida), se le agrega una tercera que es cuando se le quita el procesador al proceso que lo tiene, a pesar de que necesitaba más tiempo. Los motivos para quitarle el procesador pueden ser varios, por ejemplo: se le había asignado el procesador por un tiempo y ese tiempo terminó,llegó un proceso de mayor prioridad, etc.

44. ¿Qué algoritmos se usan para planificar hilos?

- Algoritmos
    i. Primero en entrar; Primero en salir (FIFO)
    ii. Por prioridad
    iii. Trabajo más corto (SJF)
    iv. Por Torneo (Round Robin)
    v. Colas múltiples (MQ)
    vi. Tiempo real (RT)

Referencias:

- Carretero Pérez, J., De Miguel Anasagasti, P., García Carballeira, F., & Pérez Costoya, F. (2001). Sistemas operativos. Una visión aplicada. Mac Graw Hill.
- Tanenbaum, A. S., & Bos, H. (2015). Modern operating systems. Pearson.
- Tanenbaum, A. S., & Woodhull, A. S. (1997). Operating systems: design and implementation (Vol. 68).
- Wolf, G. (2015). Fundamentos de sistemas operativos. Lulu.com.Bibliography
- 2.1.- Concepto y objetivos de los sistemas operativos. (n.d.). Retrieved September 01, 2021, from https://ikastaroak.birt.eus/edu/argitalpen/backupa/20200331/1920k/es/DAMDAW/SI/SI02/es_DAMDAW_SI02_Contenidos/website_21_concepto_y_objetivos_de_los_sistemas_operativos.html
- CETP - Tecnólogo en Informática - Montevideo. (n.d.). Retrieved September 18, 2021, from https://www.fing.edu.uy/tecnoinf/mvd /
- Dispositivos físicos del ordenador - Las nuevas tecnologías en el mundo. (n.d.). Retrieved September 01, 2021, from https://sites.google.com/site/lasnuevastecnologiasenelmundo/dispositivos-fisicos-del-ordenador
- Hilo(informatica). (2017, November 08). Retrieved September 18, 2021, from https://es.wikipedia.org/wiki/Hilo(informatica)
- López, A. (2019, May 12). CNB Bachillerato en Ciencias y Letras con orientación en Educación de Productividad y Desarrollo. Retrieved September 18, 2021, from https://studylib.es/doc/8798002/cnb-bachillerato-en-ciencias-y-letras-con-orientación-en-.. .
- Memoria compartida. (2020, November 16). Retrieved September 18, 2021, from https://es.wikipedia.org/wiki/Memoria_compartida
- Objetivo de la Asignatura. (n.d.). Retrieved September 18, 2021, from https://www.fing.edu.uy/inco/cursos/sistoper/
- Planificación de procesos en un sistema operativo. (n.d.). Retrieved September 18, 2021, from https://www.ecured.cu/Planificación_de_procesos_en_un_sistema_operativo
- Pozo, P. D. (2016, September 28). Sistemas Operativos: Semáforos. Retrieved September 18, 2021, from https://danielpozoblog.wordpress.com/2016/09/28/sistemas-operativos-semaforos/
- Problema Productor-Consumidor. (2016, June 24). Retrieved September 18, 2021, from https://es.wikipedia.org/wiki/Problema_Productor-Consumidor
- Programas de sistema - sistemasoperativosCM. (n.d.). Retrieved September 01, 2021, from https://sites.google.com/site/sistemasoperativoscm/componentes/programas-de-sistema
- ¿Qué es un sistema operativo?: Desarrollar Inclusión. (n.d.). Retrieved September 01, 2021, from https://desarrollarinclusion.cilsa.org/tecnologia-inclusiva/que-es-un-sistema-operativo/
- Salvador Ros Muñoz, J. M. (2009, February 13). Sistemas Operativos II. Retrieved September 18, 2021, from http://ocw.innova.uned.es/ocwuniversia/Ing_tecnico_infor_sistemas/SO_II/
- Sistema operativo. (2021, August 13). Retrieved September 01, 2021, from https://es.wikipedia.org/wiki/Sistema_operativo
- Sistemas Operativos - Sistemas Operativos. (n.d.). Retrieved September 01, 2021, from https://sites.google.com/site/osupaep2010/sistemas-operativos
- UNIDAD 6 - SISTEMAS_OPERATIVOS_BERE. (n.d.). Retrieved September 18, 2021, from https://sites.google.com/site/sistemasoperativosbere/home/unidad-6
- Unknown. (1970, January 01). CREACIÓN Y TERMINACIÓN DE PROCESOS. Retrieved September 18, 2021, from https://procesosenelso.blogspot.com/2012/06/creacion-y-terminacion-de-procesos.html
- (n.d.). Retrieved September 01, 2021, from http://www.lulu.com/shop/gunnar-wolf/fundamentos-de-sistemas-operativos/paperback/product-22467537.html