Unidad II: Administración de Procesos y del procesador

2.1 Concepto de proceso

Un proceso no es más que un programa en ejecución, e incluye los valores actuales del contador de programa, los registros y las variables. 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.

Un proceso es un concepto manejado por el sistema operativo que consiste en el conjunto formado por:

Las instrucciones de un programa destinadas a ser ejecutadas por el microprocesador. Su estado de ejecución en un momento dado, esto es, los valores de los registros de la CPU para dicho programa. Su memoria de trabajo, es decir, la memoria que ha reservado y sus contenidos.

Otra información que permite al sistema operativo su planificación. Esta definición varía ligeramente en el caso de sistemas operativos multadillo, donde un proceso consta de uno o más hilos, la memoria de trabajo (compartida por todos los hilos) y la información de planificación. Cada hilo consta de instrucciones y estado de ejecución.

Los procesos son creados y destruidos por el sistema operativo, así como también este se debe hacer cargo de la comunicación entre procesos, pero lo hace a petición de otros procesos. El mecanismo por el cual un proceso crea otro proceso se denomina bifurcación (folk). Los nuevos procesos pueden ser independientes y no compartir el espacio de memoria con el proceso que los ha creado o ser creados en el mismo espacio de memoria.

En los sistemas operativos multihilo es posible crear tanto hilos como procesos. La diferencia estriba en que un proceso solamente puede crear hilos para sí mismo y en que dichos hilos comparten toda la memoria reservada para el proceso. En este modelo: todo software ejecutable de la computadora, lo que a menudo

incluye al sistema operativo, está organizado en una serie del proceso secuenciales, o simplemente procesos. La idea clava aquí es que un proceso es una actividad de algún tipo: tiene programa, entrada, salida y un estado. Se puede compartir un procesador entre varios procesos, usando algún algoritmo de planificación para determinar cuándo debe de trabajar en un proceso para atender a uno distinto. Jerarquías de procesos Los sistemas operativos que manejan el concepto de proceso deben contar con algún mecanismo para crear todos los procesos necesarios. En los sistemas muy sencillos, o en los diseñados para ejecutar solo una aplicación. En otros sistemas operativos existen llamadas al sistema para crear un proceso, cargar su memoria y ponerlo en ejecutar. Sea cual sea la naturaleza exacta de la llamada al sistema. Los procesos necesitan poder crear otros procesos. En MINIX, los procesos se crean con la llamada al sistema FORK (bifurcar), que crea una copia idéntica del proceso invocador. El proceso hijo también puede ejecutar FORK, así que es posible tener un árbol de proceso.

2.2 Estados y transiciones de los procesos

El principal trabajo del procesador es ejecutar las instrucciones de máquina que se encuentran en memoria principal. Estas instrucciones se encuentran en forma de programas. Para que un programa pueda ser ejecutado, el sistema operativo crea un nuevo proceso, y el procesador ejecuta una tras otra las instrucciones del mismo. En un entorno de multiprogramación, el procesador intercalará la ejecución de instrucciones de varios programas que se encuentran en memoria. El sistema operativo es el responsable de determinar las pautas de intercalado y asignación de recursos a cada proceso. Aunque cada proceso se una entidad independiente, con su propio contador de programa y estado interno, los procesos a menudo necesitan interactuar con otros procesos. Un proceso podría generar ciertas salidas que otro proceso utilizan como entradas, en el comando de Shell. Cuando un proceso se bloquea, lo que hace porque le es imposible continuar lógicamente, casi siempre porque está separando entradas que todavía no están

disponibles, también puede ser que un programa que conceptualmente está listo y en condiciones de ejecutarse sea detenido porque el sistema operativo ha decidido asignar la CPU a otro proceso durante un tiempo. Estas dos condiciones son totalmente distintas, en el primer caso, la suspensión es inherente al problema (no es posible procesar la línea de comandos del usuarios antes de que este la teclee). En el segundo caso, se trata de un tecnicismo del sistema (no hay suficiente: CPU para darle a cada proceso su propio procesador privado).

1.- Ejecutándose (usando realmente la CPU en este instante).

2.- Listo (se puede ejecutar, pero se suspendió temporalmente para dejar que otro proceso se ejecute).

3.- Bloqueo (no puede ejecutarse en tanto no ocurra algún evento externo).

Puede haber cuánto transiciones entre estos tres estados, como se muestra.

La transacción 1 ocurre cuando un proceso descubre que no puede continuar. En algunos sistemas el proceso debe ejecutar una llamada al sistema, block, para pasar al estado bloqueado. En otros sistemas, incluido MINIX, cuando un proceso lee de un conducto o de un archivo especial, (p.ej., una terminal) y no hay entradas disponibles, se bloquea automáticamente.

Las transiciones 2 y 3 son causadas por el planificador de procesos, un parte del sistema operativo, sin que el proceso se entere siquiera de ellas.

La transición 2 ocurre cuando el planificador decide que el proceso en ejecución ya se ejecutó durante suficiente tiempo y es ahora de dejar que otros procesos tengan algo de tiempo de CPU.

La transacción 3 ocurre cuando todos los demás procesos han disfrutado de una porción justa y es hora de que el primer proceso reciba otra vez la CPU para ejecutarse.

La transacción 4 ocurre cuando acontece el suceso externo que un proceso estaba esperando (como la llegada de entrada). Sin ningún otro proceso se esta ejecutando en ese instante, se dispara de inmediato la transacción 3 y el proceso comienza a ejecutarse.

En caso contrario, el proceso tal vez tenga que esperar en el estado listo durante cierto tiempo hasta que la CPU este disponible. Usando el modelo de procesos, es mucho mas fácil visualizar lo que esta sucediendo dentro del sistema.

2.3 Procesos ligeros: Hilos o hebras

El concepto de proceso engloba dos conceptos separados y potencialmente independientes: uno relativo a la propiedad de recursos y otro que hace referencia a la ejecución. Unidad que posee recursos: A un proceso se le asigna un espacio de memoria y, de tanto en tanto, se le puede asignar otros recursos como dispositivos de E/S o ficheros.

Unidad a la que se le asigna el procesador: Un proceso es un flujo de ejecución (una traza) a través de uno o más programas. Esta ejecución se entremezcla con la de otros procesos. De tal forma, que un proceso tiene un estado (en ejecución, listo, etc) y una prioridad de expedición u origen. La unidad planificada y expedida por el sistema operativo es el proceso.

En la mayoría de los sistemas operativos, estas dos características son, de hecho, la esencia de un proceso. Sin embargo, son independientes, y pueden ser tratadas como tales por el sistema operativo. Esta distinción ha conducido en los sistemas operativos actuales a desarrollar la construcción conocida como thread, cuyas traducciones más frecuentes son hilo, hebra y proceso ligero. Si se tiene esta división de características, la unidad de asignación de la CPU se conoce como hilo, mientras que a la unidad que posee recursos se le llama proceso.

Dentro de un proceso puede haber uno o más hilos de control cada uno con:
· Un estado de ejecución (en ejecución, listo, bloqueado).

· Un contexto de procesador, que se salva cuando no esté ejecutándose.

· Una pila de ejecución.

· Algún almacenamiento estático para variables locales.

· Acceso a la memoria y a los recursos de ese trabajo que comparte con los otros hilos.

2.4 Concurrencia y secuenciabilidad

La concurrencia comprende un gran número de cuestiones de diseño, incluyendo la comunicación entre procesos, comparación y competencia por los recursos, sincronización de la ejecución de varios procesos y asignación del tiempo de procesador a los procesos y es fundamental para que existan diseños como Multiprogramación, Multiproceso y Proceso distribuido

Los procesos son concurrentes si existen simultáneamente. Cuando dos o más procesos llegan al mismo tiempo a ejecutarse, se dice que se ha presentado una concurrencia de procesos. Es importante mencionar que para que dos o más procesos sean concurrentes, es necesario que tengan alguna relación entre ellos La concurrencia puede presentarse en tres contextos diferentes:
• Varias aplicaciones: La multiprogramación se creó para permitir que el tiempo de procesador de la máquina fuese compartido dinámicamente entre varios trabajos o aplicaciones activas.

• Aplicaciones estructuradas: Como ampliación de los principios del diseño modular y la programación estructurada, algunas aplicaciones pueden implementarse eficazmente como un conjunto de procesos concurrentes.

• Estructura del sistema operativo: Las mismas ventajas de estructuración son aplicables a los programadores de sistemas y se ha comprobado que algunos sistemas operativos están implementados como un conjunto de procesos. Existen tres modelos de computadora en los que se pueden ejecutar procesos concurrentes:

• Multiprogramación con un único procesador. El sistema operativo se encarga de ir repartiendo el tiempo del procesador entre los distintos procesos, intercalando la ejecución de los mismos para dar así una apariencia de ejecución simultánea.

• Multiprocesador. Es una maquina formada por un conjunto de procesadores que comparten memoria principal. En este tipo de arquitecturas, los procesos concurrentes no sólo pueden intercalar su ejecución sino también superponerla.• Multicomputadora. Es una máquina de memoria distribuida, que está formada por una serie de computadoras. En este tipo de arquitecturas también es posible la ejecución simultánea de los procesos sobre los diferentes procesadores. En general, la concurrencia será aparente siempre que el número de procesos sea mayor que el de procesadores disponibles, es decir, cuando haya más de un proceso por procesador. La concurrencia será real cuando haya un proceso por procesador. Aunque puede parecer que la intercalación y la superposición de la ejecución de procesos presentan formas de ejecución distintas, se verá que ambas pueden contemplase como ejemplos de procesos concurrentes Existen diversas razones que motivan la ejecución de procesos concurrentes en un sistema:

• Facilita la programación de aplicaciones al permitir que éstas se estructuren como un conjunto de procesos que cooperan entre sí para alcanzar un objetivo común.

• Acelera los cálculos. Si se quiere que una tarea se ejecute con mayor rapidez, lo que se puede hacer es dividirla en procesos, cada uno de los cuales se ejecuta en paralelo con los demás.

• Posibilita el uso interactivo a múltiples usuarios que trabajan de forma simultánea.

• Permite un mejor aprovechamiento de los recursos, en especial de la CPU, ya que pueden aprovechar las fases de entrada-salida de unos procesos para realizar las fases de procesamiento de otros. Así como existen las razones que motivan la ejecución de procesos concurrentes, también existen sus contras:

• Inanición e interrupción de procesos

• Ocurrencia de bloqueos

• Que dos o más procesos requieran el mismo recurso (No apropiativo)

Tipos de procesos concurrentes.

Los procesos que ejecutan de forma concurrente en un sistema se pueden clasificar como:

Proceso independiente: Es aquel que ejecuta sin requerir la ayuda o cooperación de otros procesos. Un claro ejemplo de procesos independientes son los diferentes shells que se ejecutan de forma simultánea en un sistema. Procesos son cooperantes: Son aquellos que están diseñados para trabajar conjuntamente en alguna actividad, para lo que deben ser capaces de comunicarse e interactuar entre ellos.

En ambos tipos de procesos (independientes y cooperantes), puede producirse una serie de interacciones entre ellos y pueden ser de dos tipos:

• Interacciones motivadas porque los procesos comparten o compiten por el acceso a recursos físicos o lógicos. Por ejemplo, dos procesos independientes compiten por el acceso a disco o para modificar una base de datos.

• Interacción motivada porque los procesos se comunican y sincronizan entre sí para alcanzar un objetivo común, Por ejemplo, un compilador que tiene varios procesos que trabajan conjuntamente para obtener un solo archivo de salida.

2.5 Niveles, objetivos y criterios de planificación

En épocas pasadas de los sistemas de procesamiento por lotes, con una entrada en forma de imágenes de tarjetas en una cinta magnética, el algoritmo de planificación era sencillo: solo había que ejecutar el siguiente trabajo en la cinta.

En los sistemas de multiusuario de tiempo compartido, que se combinaban con un fondo de trabajos procesados en lote, el algoritmo era más complejo. En forma invariable, existían varios usuarios en espera de servicio y podían existir también otros trabajos para ser procesados en lote. Incluso en los sistemas puros de tiempo compartido existen con frecuencia los trabajos colaterales, como el sistema de correo electrónico, que a menudo se ejecuta todo el tiempo para enviar o recibir correo o noticias.

Cuando más de un proceso es ejecutable, el Sistema Operativo debe decidir cuál de ellos deberá ejecutarse primero. Hay que tener una planificación de los procesos que quieren ejecutarse en el sistema. La planificación es una función primordial del Sistema Operativo. La mayoría de los recursos, si no es que todos, se planifican antes de que se utilicen. La asignación de procesadores físicos a los procesos hace posible que estos realicen su trabajo, y tal asignación es un problema complejo manejado por el Sistema Operativo.

2.6 Técnicas de administración del planificador

Las disciplinas de planificación pueden ser:

- **Expropiativas**
- **No expropiativas**

Se denomina planificador al software del sistema operativo encargado de asignar los recursos de un sistema entre los procesos que los solicitan. Siempre que haya

tomar una decisión, el planificador debe decidir cuál de los procesos que compiten por la posesión de un determinado recursos lo recibirá.

Los algoritmos (técnicas) tienen distintas propiedades según los criterios en los que se basen para su construcción, lo cual se refleja en qué tipo de procesos se puede ver favorecido frente a otro en la disputa del procesador. Antes de realizar la elección de un algoritmo se debe considerar las propiedades de estos frente al criterio de diseño elegido. Algunos de estos son:

a) Eficacia: Se expresa como un porcentaje del tiempo medio de utilización. Aunque puede parecer lógico intentar mantener este parámetro próximo al 100%, con un valor tan elevado otros aspectos importantes de medida del comportamiento del sistema pueden verse deteriorados, como por ejemplo el tiempo medio de espera.

b) Rendimiento: Es una medida del número de procesos completados por unidad de tiempo. Por ejemplo 10 procesos por segundo.

c) Tiempo de retorno o regreso: Es el intervalo de tiempo que transcurre desde que un proceso se crea o presenta hasta que completa por el sistema.

d) Tiempo de espera: Es el tiempo que el proceso espera hasta que se le

concede el procesador. Puede resultar una medida más adecuada de la eficiencia del sistema, ya que se elimina de la media el tiempo que tarda en ejecutarse el mismo.

e) Tiempo de respuesta a un evento: Se denomina así el intervalo de tiempo que transcurre desde que se señala un evento hasta que se ejecuta la primera instrucción de la rutina de servicio de dicho evento. El criterio de selección de un algoritmo se suele basar en la maximización o minimización de una función de los parámetros anteriores.