Estados de un proceso

traza

El comportamiento de un proceso individual puede caracterizarse por la lista de la secuencia de instrucciones que se ejecutan para dicho proceso.

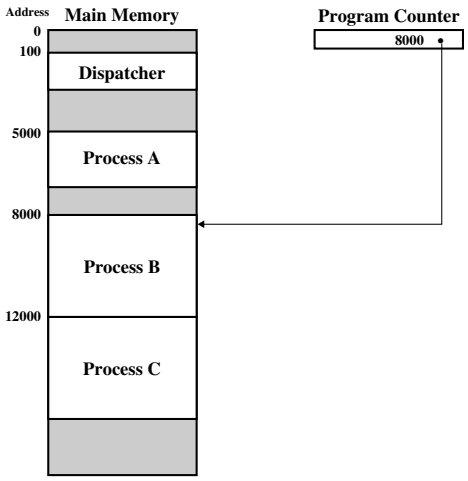


Figure 3.1 Snapshot of Example Execution (Figure 3.3) at Instruction Cycle 13

(a) Trace of Process A	(b) Trace of Process B	(c) Trace of Process C
5000	8000	12000
5001	8001	12001
5002	8002	12002
5003	8003	12003
5004		12004
5005		12005
5006		12006
5007		12007
5008		12008
5009		12009
5010		12010
5011		12011

5000 = Starting address of program of Process A
8000 = Starting address of program of Process B
12000 = Starting address of program of Process C

Figure 3.2 Traces of Processes of Figure 3.1

1	5000	27	12004
2	5001	28	12005
3	5002		Time out
4	5003	29	100
5	5004	30	101
6	5005	31	102
	Time out	32	103
7	100	33	104
8	101	34	105
9	102	35	5006
10	103	36	5007
11	104	37	5008
12	105	38	5009
13	8000	39	5010
14	8001	40	5011
15	8002		Time out
16	8003	41	100
	I/O request	42	101
17	100	43	102
18	101	44	103
19	102	45	104
20	103	46	105
21	104	47	12006
22	105	48	12007
23	12000	49	12008
24	12001	50	12009
25	12002	51	12010
26	12003	52	12011
			Time out

100 = Starting address of dispatcher program

shaded areas indicate execution of dispatcher process;
first and third columns count instruction cycles;
second and fourth columns show address of instruction being executed

Figure 3.3 Combined Trace of Processes of Figure 3.1

Un modelo de dos estados

La principal responsabilidad del sistema operativo es controlar la ejecución de los procesos; esto incluye la terminación de las pautas de intercalado que se van a seguir y la asignación de recursos a los procesos. Un proceso puede estar en uno de dos estados.

- Ejecución.
- No ejecución.

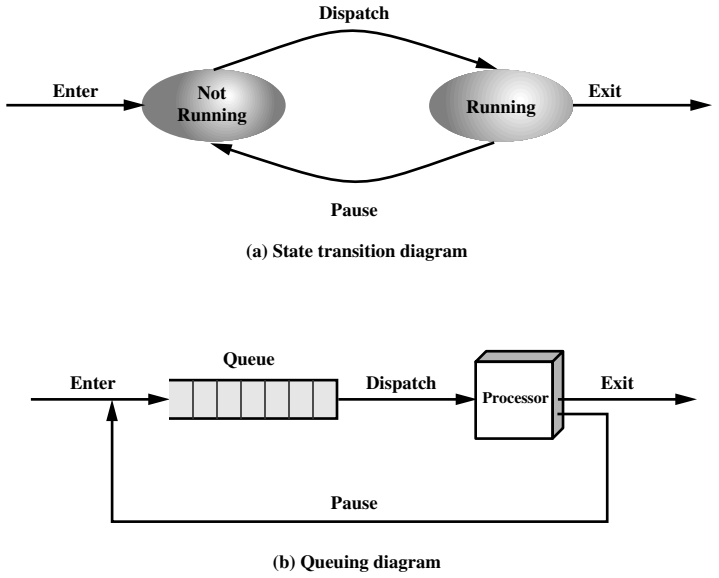


Figure 3.4 Two-State Process Model

Creación y terminación de procesos

La vida de un proceso está limitada por su creación y su terminación.

Creación

Nuevo trabajo por lotes	El sistema operativo está provisto de un flujo de control de trabajo por lotes, generalmente en cinta o en disco. Cuando el sistema operativo se prepara para coger un nuevo trabajo, leerá la próxima secuencia de órdenes de control de trabajos.
Conexión interactiva	Un usuario entra en el sistema desde un terminal.
Creador por el SO para dar un servicio	El sistema operativo puede llevar a cabo una función de parte de un programa de usuario, sin que el usuario tenga que esperar (por ejemplo, un proceso para control de impresión).
Generado por un proceso existente	Para modular o para aprovechar el paralelismo, un programa de usuario puede ordenar la creación de una serie de procesos.

Terminación de procesos

Terminación normal	El proceso ejecuta una llamada a un servicio del SO que indica que ha terminado de ejecutar.
Tiempo límite excedido	El proceso ha ejecutado por más tiempo del límite total especificado. Hay varias posibilidades para la clase de tiempo que se mide. Entre éstas se incluyen el tiempo total transcurrido («tiempo de reloj de pared»), el tiempo que ha estado ejecutando y, en el caso de un proceso interactivo, el tiempo transcurrido desde que el usuario realizó su última entrada de datos.
No hay memoria disponible	El proceso requiere más memoria de la que el sistema le puede proporcionar.
Violación de límites	El proceso trata de acceder a una posición de memoria a la que no le está permitido acceder.
Error de protección	El proceso intenta utilizar un recurso o un archivo que no le está permitido utilizar o trata de utilizarlo de forma incorrecta.
Error aritmético	El proceso intenta hacer un calculo prohibido, como una división por cero o trata de almacenar un número mayor del que el hardware acepta.
Tiempo máximo de espera rebasado	El proceso ha esperado más allá del tiempo máximo especificado para que se produzca cierto suceso.

Terminación de procesos

Fallo de E/S	Se produce un erro en la entrada o la salida, como la incapacidad de encontrar un archivo, un fallo de lectura o escritura después de un número máximo de intentos.
Instrucción ilegal	El proceso intenta ejecutar una instrucción inexistente (a menudo como resultado de un salto a una zona de datos para intentar ejecutar los datos).
Instrucción privilegiada	El proceso intenta usar una instrucción reservada para el sistema operativo.
Mal uso de los datos	Un elemento de datos es de un tipo equivocado o no está iniciado.
Intervención del operador o del SO	Por alguna razón, el operador o el sistema operativo termina con el proceso.
Terminación del padre	Cuando un proceso padre finaliza, el sistema operativo puede diseñarse para terminar automáticamente con todos sus descendientes.
Solicitud del padre	Un proceso padre tiene normalmente la autoridad de terminar con cualquiera de sus descendientes.

Figure 3.5 Five-State Process Model

Un modelo de cinco estados

Ejecución: el proceso que está actualmente en ejecución.

Listo: proceso que esta preparado para ejecutarse, en cuanto se le dé la oportunidad.

Bloqueado: proceso que no se puede ejecutar hasta que se produzca cierto suceso, como la terminación de una operación de E/S.

Nuevo: proceso que se acaba de crear, pero aún no ha sido admitido por el sistema operativo en el grupo de procesos ejecutables. Normalmente, un proceso *Nuevo* aún no está cargado en la memoria principal.

Terminado: un proceso que ha sido excluido por el sistema operativo del grupo de procesos ejecutables, bien porque se detuvo o porque fue abandonado por alguna razón.

Transiciones en el modelo de cinco estados

- Nulo → Nuevo
- Nuevo → Listo
- Listo → Ejecución
- Ejecución → Terminado
- Ejecución → Listo
- Ejecución → Bloqueado
- Bloqueado → Listo
- Listo → Terminado
- Bloqueado → Terminado

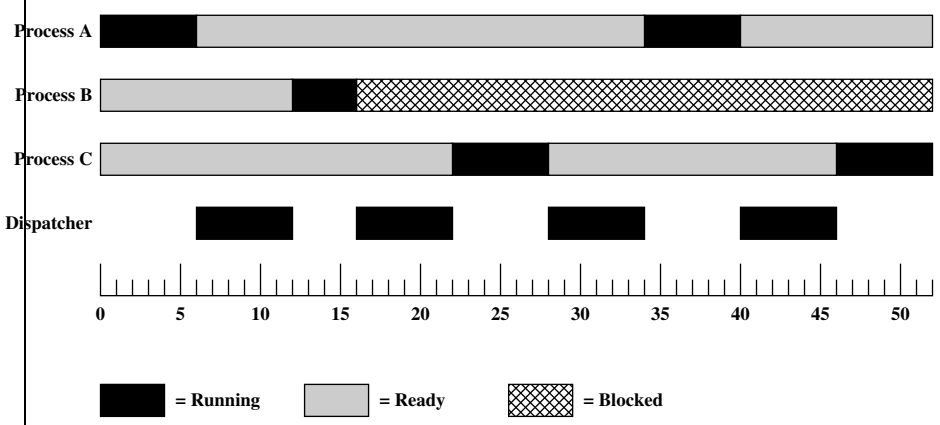


Figure 3.6 Process States for Trace of Figure 3.3

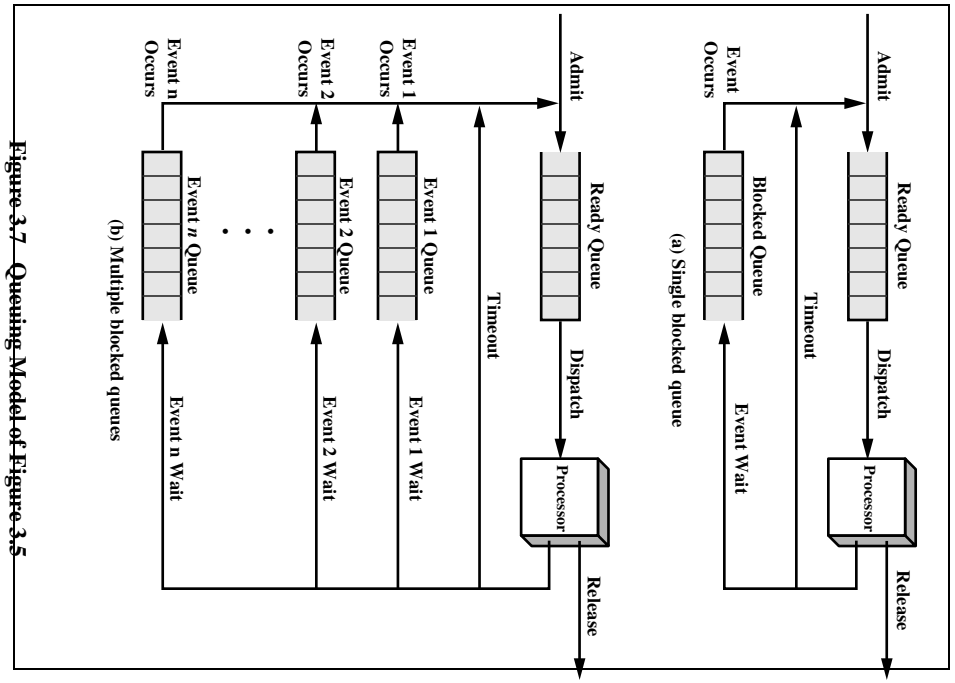


Figure 3.7 Queuing Model of Figure 3.5

Procesos suspendidos

Si el sistema no utiliza memoria virtual, el proceso que debe ejecutarse debe ser cargado por completo en la memoria principal. Se *debería* tener la suficiente memoria principal para alojar todos los procesos posibles y mantener sus requerimientos de memoria.

El *intercambio* significa mover una parte del proceso a todo el proceso de la memoria principal al disco. Cuando ninguno de los procesos en la memoria principal están en estado *Listo*, el sistema operativo pasa al disco uno de los procesos que esté *Bloqueado* y lo lleva a una cola de *Suspendidos*.

Razones para el intercambio

Intercambio	El sistema operativo necesita liberar suficiente memoria principal para cargar un proceso que está listo para ejecutarse.
Otra razón del SO	El sistema operativo puede suspender a un proceso subordinado, o a un proceso que se sospecha que sea el causante de un problema.
Solicitud de un usuario interactivo	Un usuario puede querer suspender la ejecución de un programa con fines de depuración o en conexión con el uso de un recurso.
Temporización	Un proceso puede ejecutarse periódicamente y puede ser suspendido mientras espera el siguiente intervalo de tiempo.
Solicitud del proceso padre	Un proceso padre puede querer suspender la ejecución de un descendiente para examinar o modificar el proceso suspendido o para coordinar la actividad de varios descendientes.

Modelo de estados de los procesos

Nuevo

Listo

Bloqueado

Ejecución

Bloqueado y suspendido

Listo y suspendido

Terminado

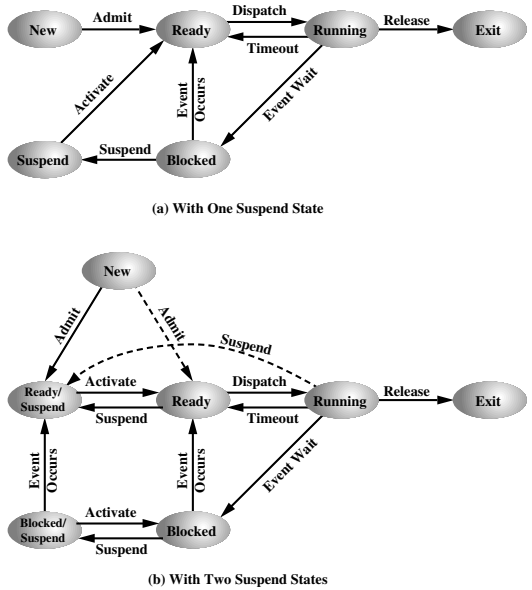


Figure 3.8 Process State Transition Diagram with Suspend States

Nuevas transiciones al modelo

- Bloqueado → Bloqueado y suspendido
- Bloqueado y suspendido → Listo y suspendido
- Listo y suspendido → Listo
- Listo → Listo y suspendido
- Nuevo → Listo y suspendido, Nuevo → Listo
- Bloqueado y suspendido → Bloqueado
- Ejecución → Listo y suspendido
- Varios → Terminado

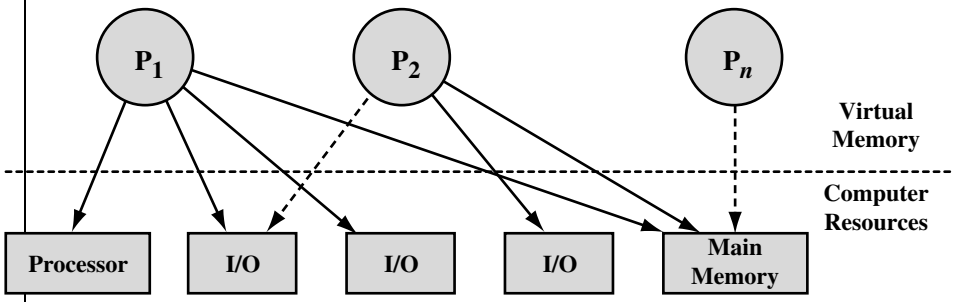


Figure 3.9 Processes and Resources (resource allocation at one snapshot in time)

Descripción de procesos

Tablas de memoria

- La localización del proceso en memoria principal
- La localización del proceso en memoria secundaria.
- Cualquier atributo de protección de los bloques de la memoria principal o de la memoria virtual, tales que los procesos puedan acceder a ciertas regiones de memoria compartida.
- Cualquier información necesaria para gestionar la memoria virtual.

Tablas de entrada y salida

Son utilizadas por el sistema operativo para administrar los dispositivos de entrada y salida y los canales del sistema de computador.

Tablas de archivos

Suministra información acerca de la existencia de archivos, su localización en memoria secundaria, su estado actual y otros atributos.

Tablas de procesos

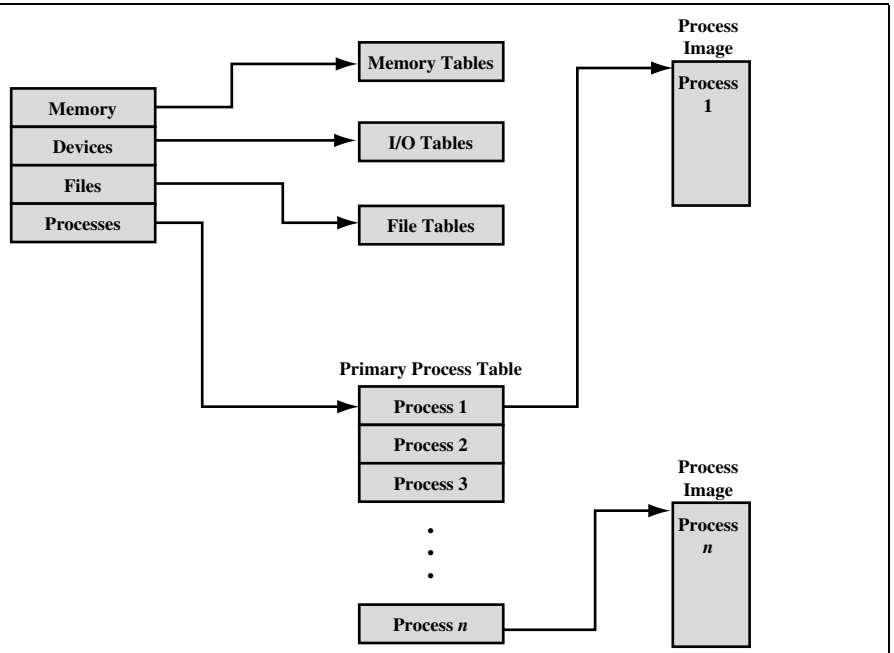


Figure 3.10 General Structure of Operating System Control Tables

Estructuras de control de los procesos

El sistema operativo debe saber dos cosas para gestionar los procesos, en primer lugar, saber donde el proceso está localizado y segundo, este debe saber los atributos del proceso que son necesarios para su gestión.

Imagen del proceso

Datos de usuario. La parte modificable del espacio de usuario. Puede incluir datos del programa, área de pila de usuario y los programa que pueden ser modificados.

Programa de usuario. El programa a ser ejecutado.

Pila del sistema. Cada proceso tiene una o varias pilas (LIFO) asociadas con él para las llamadas al sistema. Una pila es utilizada para almacenar los parámetros y las direcciones de retorno de los procedimientos y llamadas al sistema.

Bloque de control del proceso. Datos necesitados por el proceso para controlar el proceso.

Localización del proceso

- Depende del esquema de gestión de memoria que esta siendo utilizado.
- En el caso más simple, son mantenidos como como bloques de memoria contiguos y continuos.
- Este bloque es mantenido en memoria secundaria, usualmente en disco.
- Para ejecutar el proceso, la imagen completa del proceso debe ser mantenido en memoria principal o la menos en memoria virtual.

- **Identificación de proceso**. Los identificadores numéricos que puede ser almacenados en el bloque de control del proceso incluyen:
  - Identificador del proceso.
  - Identificador del proceso que creo el proceso (padre).
  - Identificador de usuario.
- **Información del estado del proceso**
  - Registros visibles de usuario. Un registro visible es un registro que puede ser referenciado por medio del lenguaje de máquina que el procesador ejecuta.
  - Registros de control y estado. Estos son una variedad de registros que son empleados para controlar la operación del procesador.
    - *Contador de programa.* Contiene la dirección de la siguiente instrucción a ser alcanzada.
    - *Códigos de condición.* Resulta de la más reciente operación aritmética u operación lógica.
    - *Información de estado.* Incluye las banderas de habilitación y des-habilitación de interrupciones, modo de ejecución.
  - Apuntadores a pila

Figure 3.11 Pentium II EFLAGS Register

Atributos de proceso

- Identificación de procesos.
- Información de estado del procesador.
- Información de control del proceso.

- **Información de control del proceso**
  - Información de estado y planificación.
    - *Estado del proceso.* Define el estado actual del proceso.
    - *Prioridad.* Uno o más campos puede ser utilizados para describir la prioridad de planificación del proceso.
    - *Información relacionada a la planificación.* Depende del algoritmo de planificación utilizado.
    - *Evento.* Identificación del evento que un proceso esta esperando para ser reasumido.
  - Estructura de datos. Un proceso puede estar enlazado con otro proceso en una cola, un anillo u otra estructura.
  - Comunicación entre procesos. Varias banderas, señales y mensajes pueden estar asociados con la comunicación entre dos proceso independientes.
  - Privilegios del proceso. A los procesos se les concede privilegios en términos de la memoria que puede ser accedida y los tipos de instrucciones que pueden ser ejecutadas.
  - Gestión de memoria. Esta sección puede incluir apuntadores a los segmentos de memoria o a las tablas de páginas que describen la memoria virtual asignada a este proceso.
  - Propiedad de recursos y utilización. Los recursos son controlados por el proceso pueden ser indicados.

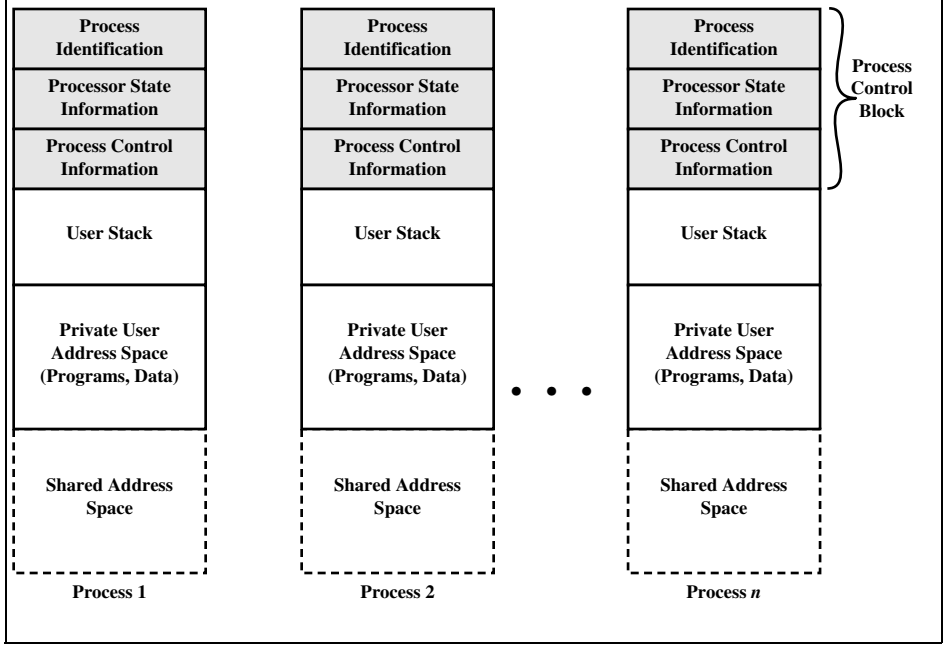


Figure 3.12 User Processes in Virtual Memory

El rol del bloque de control de procesos

El bloque de control de proceso es la estructura más importante dentro del sistema operativo. Cada bloque de control contiene toda la información acerca de un proceso que es necesaria por el SO. Los bloques son leído y modificados por virtualmente cada módulo en el sistema operativo.

Inconvenientes

- Un *bug* en una sola rutina puede dañar el bloque de control de procesos.
- Un cambio de diseño en la estructura o semántica del bloque de control de proceso puede afectar un gran número de módulos dentro del del SO.

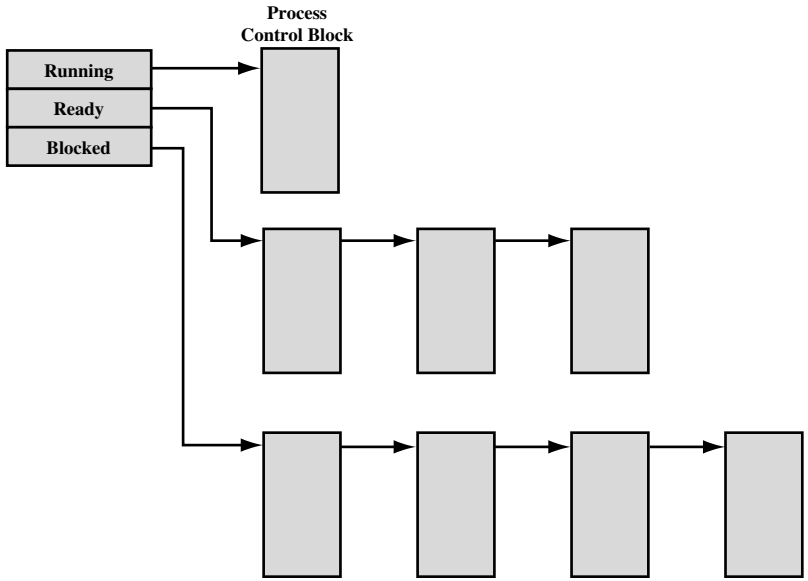


Figure 3.13 Process List Structures

Control del proceso

- Modos de ejecución.
- Creación de procesos.
- Conmutación de procesos.
- Ejecución del sistema operativo.

Creación de procesos

1. Asignación de un identificador de proceso al nuevo proceso.
2. Localizar espacio para el proceso.
3. Inicializar el bloque de control del proceso.
4. Establecer los enlaces apropiados.
5. Crear y expandir otras estructuras de datos.

Conmutación de procesos

Una conmutación de procesos puede ocurrir en cualquier momento que el sistema operativo ha obtenido el control del proceso actualmente en ejecución.

- Interrupción del reloj.
- Interrupción de entrada/salida.
- Fallo de página.

Modo de conmutación

1. Guardar el contexto del programa actual corriendo.
2. Establece el contador de programa a la dirección inicial de programa que maneja la interrupción.
3. Conmuta de modo usuario a modo kernel así el código de procesamiento de interrupción puede incluir instrucciones privilegiadas.

Cambio del estado del proceso

1. Guardar el contexto del procesador, incluyendo el contador de programa y otros registros.
2. Actualizar el bloque de control del proceso que esta actualmente en modo de ejecución. Esto incluye cambiar el estado del proceso a uno de los otros estados.
3. Mover el bloque de control del proceso de este proceso a la cola apropiada.
4. Seleccionar otro proceso para ejecución.
5. Actualizar el bloque de control del proceso seleccionado. Esto incluye cambiar el estado de este proceso a ejecución.
6. Actualizar las estructuras de datos que gestionan la memoria.
7. Restablecer el contexto del procesador que existía en el momento que el proceso seleccionado fue cambio del estado de ejecución, cargando los anteriores valores del contador de programa y otros registros.

Ejecución del sistema operativo

- El sistema operativo funciona de la misma forma que el *software* de computador corriente, esto es, un programa ejecutado por el computador.
- El sistema operativo frecuentemente cede el control y depende del procesador restablezca el control al sistema operativo.

Modos

- Kernel separado.
- Funciones del SO ejecutadas dentro del proceso usuario.
- Funciones del SO ejecutadas como un proceso separado.

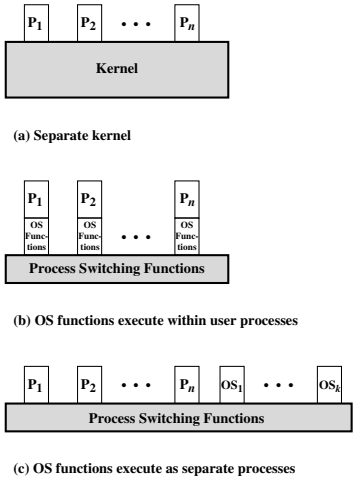


Figure 3.14 Relationship Between Operating System and User Processes

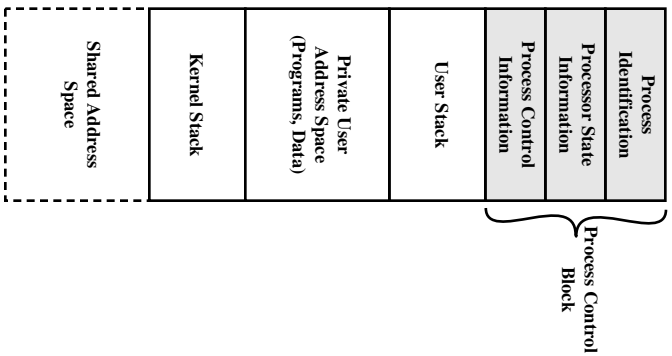


Figure 3.15 Process Image: Operating System Executes Within User Space

Procesos e hilos

El concepto de proceso encarna dos características:

Propietario de recursos: Un proceso incluye un espacio de direcciones virtuales para mantener la imagen del proceso y a medida que transcurre su ejecución mantiene un conjunto de recursos.

Ejecución/Planificación: La ejecución de un proceso un camino de ejecución de uno o varios programas. Esta ejecución puede ser separada de otro proceso. Un proceso tiene un prioridad, un estado de ejecución y es la entidad que es despachada y planificada por el sistema operativo.

Dentro de un proceso, pueden haber uno o más hilos de ejecución, cada uno con lo siguiente:

- Un estado de ejecución del hilo.
- Un contexto del hilo guardado cuando no esta corriendo.

Multi-hilos

Se refiere a la habilidad de un sistema operativo soportar la ejecución de múltiples hilos de ejecución dentro de un proceso.
En un ambiente multi-hilo, un proceso es definido como la unidad de localización de recursos y una unidad de protección.

- Un espacio virtual de direcciones que mantiene la imagen del proceso.
- Acceso protegido al procesador, otros procesos, archivos y recursos de E/S.

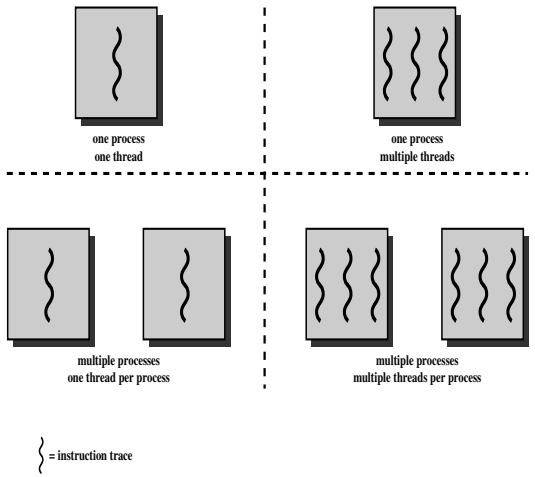


Figure 4.1 Threads and Processes [ANDE97]

Información de un hilo

Dentro de un proceso, puede existir uno o más hilos, cada uno con lo lo siguiente:

- Un estado de ejecución de un hilo (Corriendo, Listo, etc.).
- Un contexto de hilo salvado cuando no está corriendo; una forma de ver un hilo es como un contador de programa independiente corriendo dentro de un proceso.
- Un pila de ejecución.
- Algún almacenamiento estático por hilo para variables globales.
- Acceso a la memoria y a los recursos de sus procesos, compartido con los otros hilos en ese proceso.

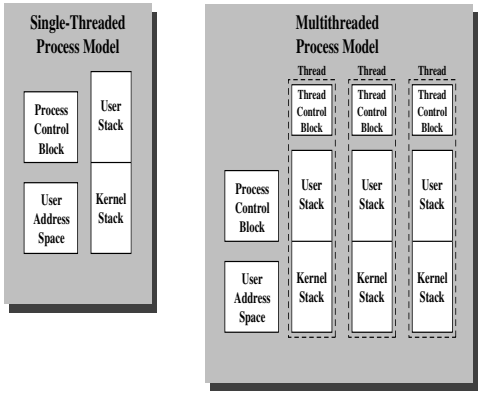


Figure 4.2 Single Threaded and Multithreaded Process Models

Los principales beneficios de los hilos

Los principales beneficios de los hilos derivan de las siguientes implicaciones de desempeño:

1. Toma menos tiempo crear un nuevo hilo en un proceso existente que crear un nuevo proceso.
2. Toma menos tiempo terminar un hilo que un proceso.
3. Toma menos tiempo conmutar entre dos hilos dentro del mismo proceso.
4. Los hilos mejoran la eficiencia de comunicación entre diferentes programas en ejecución.

   Si una aplicación o una función debe ser implementada como un conjunto de unidades relacionadas de ejecución, es más eficiente hacerlo como una colección de hilos que una colección de procesos separados.

Funcionalidad de hilos

Como los procesos, los hilos tiene *estados de ejecución* y se puede *sincronizar* con otros.

Estados de los hilos

Como con los procesos, los estados principales para un hilo son Corriendo, Listo y Bloqueado. No tiene sentido hablar del estado suspendido, por que este es un concepto de proceso.

Operaciones asociadas con el cambio de estado de un proceso

- Creación. Cuando un proceso es creado, también es creado un hilo. Un hilo puede crear a su vez otros hilos.
- Bloquear. Cuando un hilo necesita esperar por un evento, este se bloqueará.
- Desbloquear. Cuando el evento por el cual un hilo es bloqueado ocurre, el hilo es movido a la cola de listo.
- Terminar. Cuando un hilo es completado, su registro de contexto y pilas son liberados.

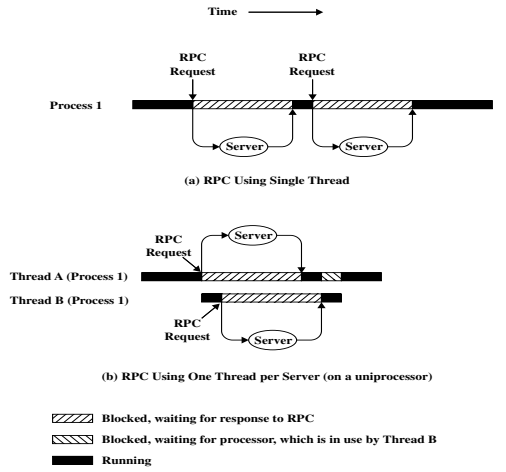


Figure 4.3 Remote Procedure Call (RPC) Using Threads

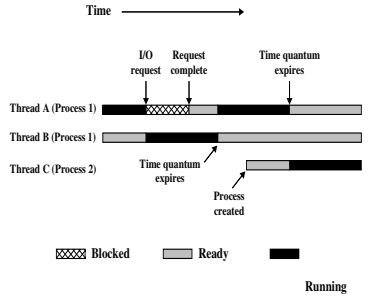


Figure 4.4 Multithreading Example on a Uniprocessor

Sincronización de hilos

- Todos los hilos comparten *el mismo espacio de direcciones* y *los recursos*.
- Cualquier alteración de un recurso por un hilo afecta el ambiente de otros hilos en el mismo proceso.
- Lo anterior implica la necesidad de sincronizar las actividades de varios hilos.

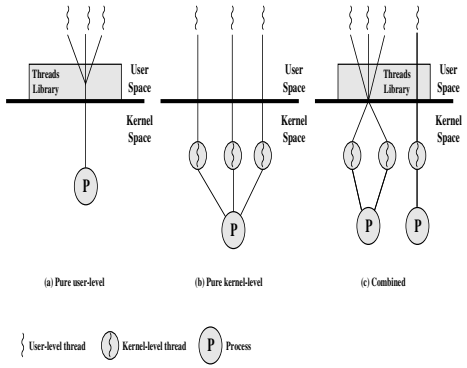


Figure 4.6 User-Level and Kernel-Level Threads

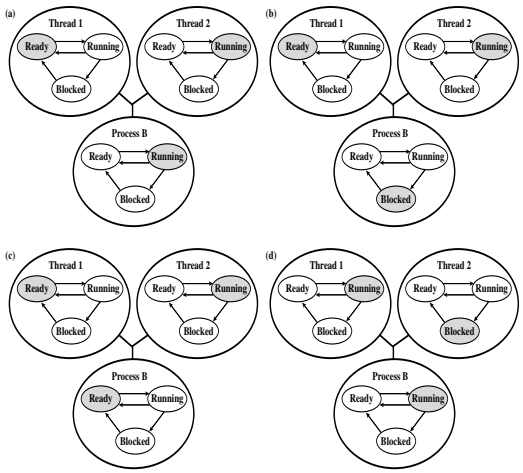


Figure 4.7 Examples of the Relationships Between User-Level Thread States and Process States

Ventajas y desventajas de HNU y HNK

Ventajas

1. La conmutación no requiere privilegios a nivel de kernel.
2. La planificación puede ser específica a la aplicación.
3. HNU pueden corren en cualquier sistema operativo.

Desventajas

1. Muchas llamadas al sistema son bloqueantes por lo tanto pueden conducir a bloquear a todo el proceso.
2. En una estrategia pura HNU, una aplicación multi-hilos no puede tomar ventaja de un sistema multiprocesador.

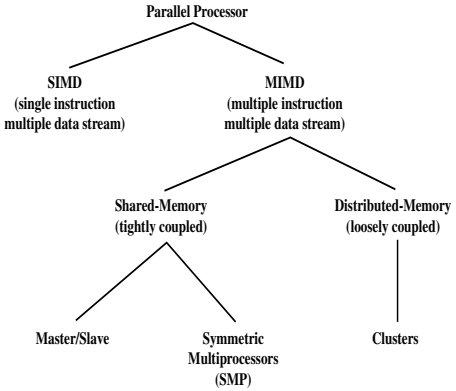


Figure 4.8 Parallel Processor Architectures

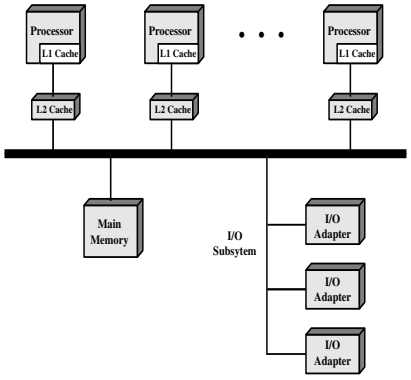


Figure 4.9 Symmetric Multiprocessor Organization

Consideraciones de diseño con sistemas SMP

Hilos o procesos simultáneamente concurrentes

Planificación

Sincronización

Administración de memoria

Confiabilidad y tolerancia a fallos

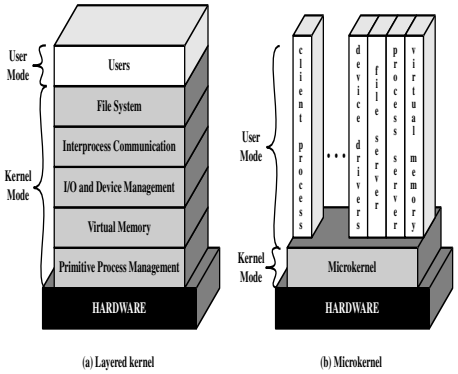


Figure 4.10 Kernel Architecture

Beneficios de la organización de los microkernels

- Interfaces uniformes.
- Extensiblidad.
- Flexibilidad.
- Portabilidad.
- Confiabilidad.
- Soporte para sistemas distribuidos.
- Soporte para sistemas operativos orientados a objetos (OOOS).

Diseño del microkernel

Administración de memoria de bajo nivel

- Conceder.
- Mapear.
- Reclamar.

Comunicación entre procesos
Administración de interrupciones y E/S

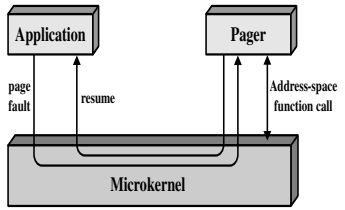


Figure 4.11 Page Fault Processing

Tipos de planificación

Planificación a largo-plazo	Decisión de añadir procesos al conjunto de procesos a ejecutar.
Planificación a mediano-plazo	Decisión de añadir procesos al conjunto de procesos que se encuentran parcial o completamente en la memoria.
Planificación de corto-plazo	Decisión sobre qué proceso disponible será ejecutado en el procesador.
Planificación de entrada-salida	Decisión sobre qué solicitud de E/S pendiente será tratada por un dispositivo de E/S disponible.

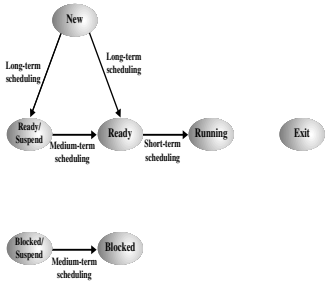


Figure 9.1 Scheduling and Process State Transitions

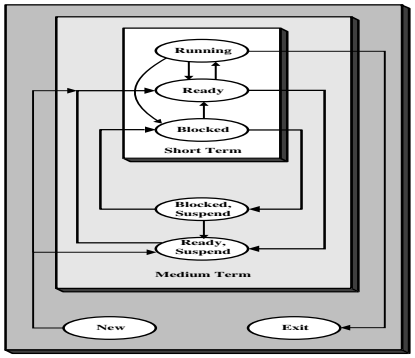


Figure 9.2 Levels of Scheduling

Planificación a largo plazo

Determina cuáles son los programas admitidos por el sistema.

Grado de multiprogramación

- Cuanto más procesos se crean, menor es el porcentaje de tiempo en el que cada proceso se puede ejecutar.
- Si el procesador está desocupado se excede o cierto umbral se puede invocar al planificador de largo plazo.

Decisión que programas aceptar

- Varios criterios: FCFS (First Come, First Server) o una una herramienta de gestión.
- Prioridades, tiempos de ejecución y exigencias de E/S.
- Equilibrar el procesamiento con E/S.
- Interactivos. Todas la solicitudes son aceptadas hasta saturar el sistema.

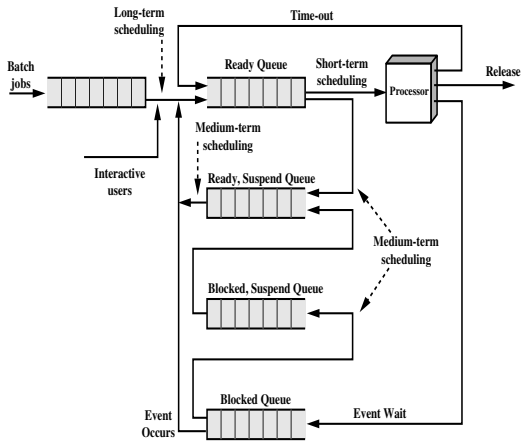


Figure 9.3 Queuing Diagram for Scheduling

Planificación a mediano plazo

La idea clave detrás de este planificador es que puede ser ventajoso remover procesos de la memoria (y de una disputa activa por la CPU) y de esta manera reducir el grado de multiprogramación.

En algún momento posterior, el proceso puede ser introducido de nuevo a la memoria y continuar su ejecución desde el punto que se suspendió.

El planificador de mediano plazo remueve el proceso **swap out** y posteriormente lo introduce **swap in**.

Planificación de corto plazo

También conocido como *dispatcher*. Actúa cuando se produce un suceso que puede conducir a la interrupción del proceso actual.

- Interrupción del reloj.
- Interrupción de E/S.
- Llamadas al sistema operativo.
- Señales.

Algoritmos de planificación

"El principal objetivo de la planificación a corto plazo es repartir el tiempo del procesador de forma que se optimicen uno o más elementos del comportamiento del sistema."

Generalmente se fija un conjunto de *criterios*.

- Orientados al usuario. Al comportamiento del sistema tal y como lo persiben los usuarios.
- Orientados al sistema. El uso efectivo y eficiente del procesador.

Hay criterios *cuantitativos* y *cualitativos*

Utilización de la CPU . Se quiere mantener la CPU tan ocupada como sea posible. El grado de utilización puede estar entre un 0 y 100 %.

Rendimiento. Si la CPU está ocupada ejecutando procesos, entonces se está realizando trabajo. La unidad de medida es el número de procesos que se completan por unidad de tiempo (*throughput*).

Tiempo de entrega (o retorno). Es el intervalo de tiempo transcurrido entre el lanzamiento de un proceso y su finalización.

Tiempo de espera. El tiempo de espera es la suma de los periodos esperando en la cola de listos.

Tiempo de respuesta. Para un proceso interactivo, es el intervalo de tiempo transcurrido desde que se emite una solicitud hasta que se comienza a recibir la respuesta.

Uso de prioridades

En vez de una sola cola de listo, se ofrece un conjunto de colas en orden de prioridad descendente: CL_0, CL_1, $\ldots CL_N$ donde la prioridad $P[CL_i] > P[CL_j]$ para $i < j$.

Figure 9.4 Priority Queuing

Otras políticas de planificación

La **función de selección** determina que proceso, de entre los listos, se elige para ejecutar a continuación. La función puede estar basada en prioridades, necesidades de recursos o en las características de ejecución de los procesos.
El **modo de decisión** especifica los instantes de tiempo en que se aplica la función de selección.

No expropiativo. Cuando un proceso conmuta del estado de ejecución al estado de espera (por ejemplo, en una solicitud de E/S, o al invocar una espera para la terminación de uno de los procesos hijos). Cuando un proceso termina.

Expropiativo. Cuando un proceso cambia de estado de ejecución al estado de listo (por ejemplo cuando ocurren una interrupción). Cuando un proceso pasa del estado de espera al estado de listo (por ejemplo, en la terminación de una operación de E/S).

	Función de selección	Modo de decisión	Productividad	Tiempo de respuesta	Sobrecarga	Efecto sobre los procesos	Inanición
FCFS	$max[w]$	No expropiativo	No relevante	Puede ser alto, especialmente si varía mucho los tiempos de ejecución	Mínima	Penaliza los procesos cortos; penaliza los procesos con carga de E/S	No
Turno rotatorio	Constante	Expropiativo (en los quatums de tiempo)	Puede ser baja si el quatum es muy pequeño	Ofrece un buen tiempo de respuesta para procesos cortos	Mínima	Trato equitativo	No
SPN	$min[s]$	No expropiativo	Alta	Ofrece un buen tiempo de respuesta para proceso cortos	Puede ser alta	Penaliza los procesos largos	Posible
SRT	$min[s - e]$	Expropiativo (en la llegada)	Alta	Ofrece un buen tiempo de respuesta	Puede ser alta	Penaliza los procesos largos	Posible
HRRN	$max\left(\frac{w+s}{s}\right)$	No expropiativo	Alta	Ofrece un buen tiempo de respuesta	Puede ser alta	Buen equilibrio	No
Fedback		Expropiativo (en los quantums de tiempo)	No relevante	No relevante	Puede ser alta	Puede favorecer a los procesos con carga de E/S	Posible

Proceso	Instante de llegada	Tiempo de servicio
A	0	3
B	2	6
C	4	4
D	6	5
E	8	2

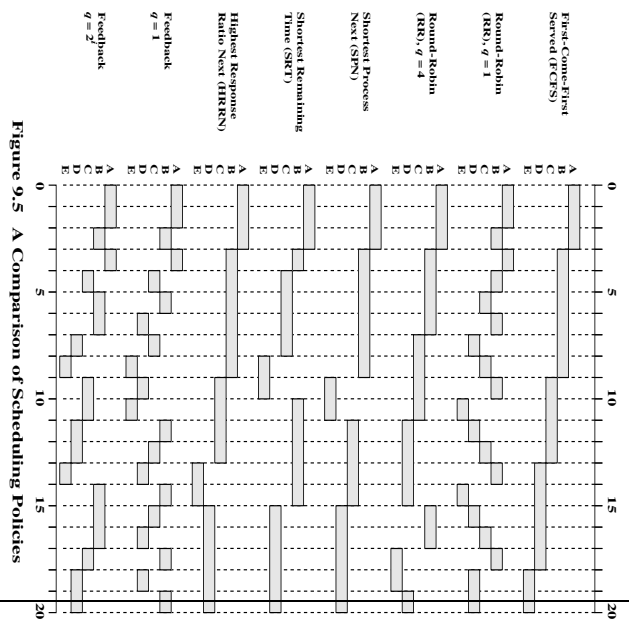
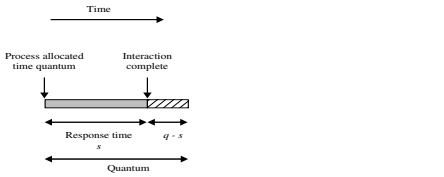
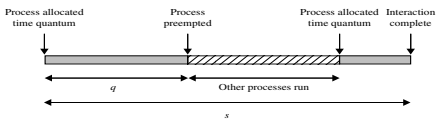


Figure 9.5 A Comparison of Scheduling Policies



(a) Time quantum greater than typical interaction



(b) Time quantum less than typical interaction

Figure 9.6 Effect of Size of Preemption Time Quantum

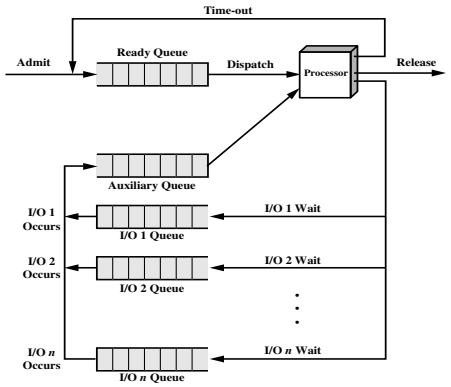


Figure 9.7 Queuing Diagram for Virtual Round-Robin Scheduler

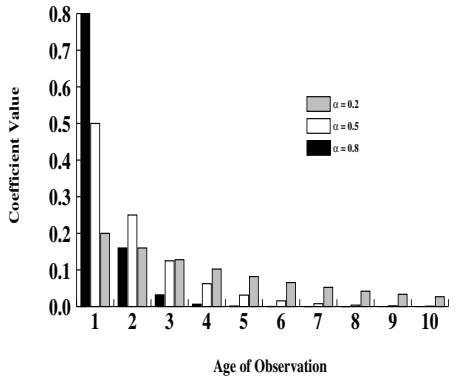


Figure 9.8 Exponential Smoothing Coefficients

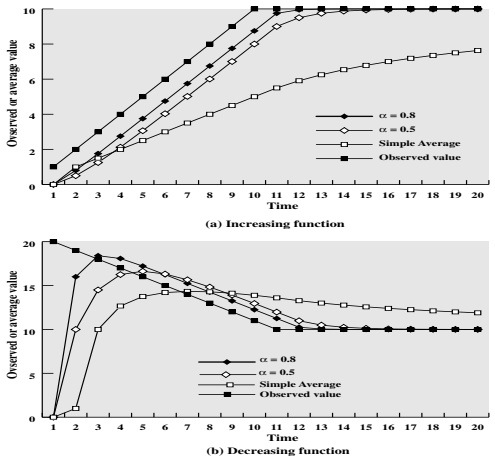


Figure 9.9 Use of Exponential Averaging

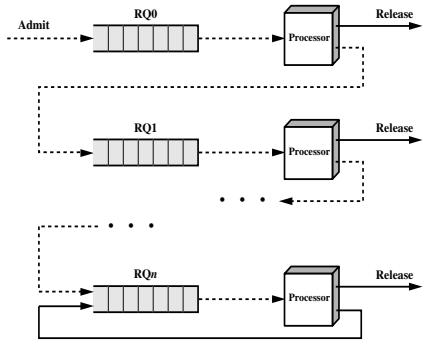
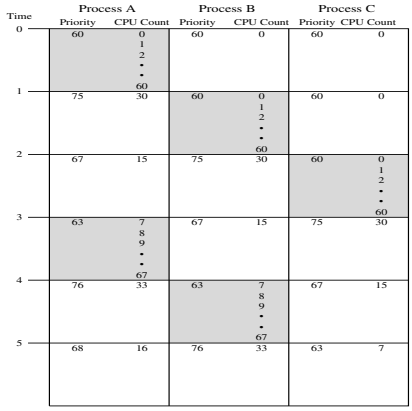


Figure 9.10 Feedback Scheduling



Shaded rectangle represents executing process

Figure 9.17 Example of Traditional UNIX Process Scheduling

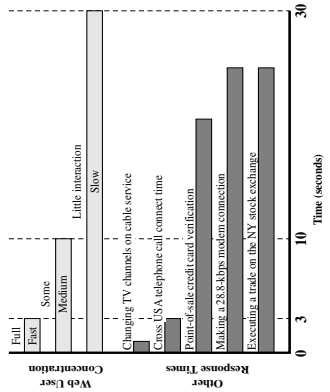
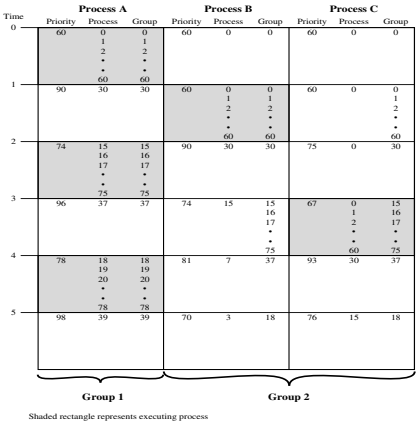


Figure 9.21 Response Time Requirements [SEVC96]



Shaded rectangle represents executing process

Figure 9.16 Example of Fair Share Scheduler—Three Processes, Two Groups

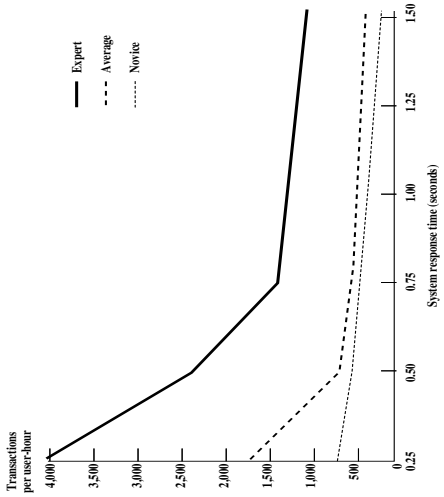


Figure 9.20 Response Time Results for High-Function Graphics

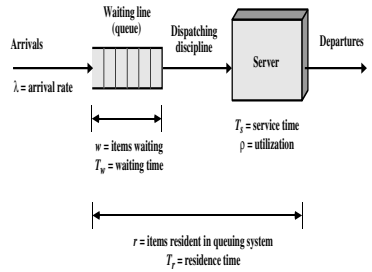


Figure 9.22 Queuing System Structure and Parameters for Single-Server Queue

Planificación de multiprocesadores

Elementos de diseño

- La asignación de los procesos a los procesadores.
- El uso de la multiprogramación en los procesadores individuales.
- La expedición real de procesos.

Asignación de procesos a los procesadores

El método de planificación más simple consiste en tratar a los procesadores como un recurso reservado y asignar a los procesos los procesdores por demanda. Existen dos métodos:

- Estática.
- Dinámica.

Uso de multiprogramación

Cuando cada proceso se asigna estáticamente a un procesador durante su ciclo de vida surge una nueva cuestión:

¿Puede estar multiprogramado dicho procesador?

Grano grueso o independiente. Alto nivel de multiprogramación.

Grano medio
- No es importante que cada porcesador esté ocupado al máximo.
- Mejor rendimiento promedio. Todos los hilos estánd disponibles para ejecutarse de inmediato.

Planificación de procesos

En la mayoría de los sistemas multiprocesador, los procesos no se asignan a los procesadores de forma dedicada.

Planificación de hilos

La potencia de los hilo se lleva a cabo en un sistema multiprocesador.

Si son de grano grueso. No hay mejoras.

Si son de grano fino. Pequeñas diferencias de planificación. La gestión tiene un impacto significativo.

Propuestas

Reparto de carga: No se asigna a un procesador en particular. Se mantiene un cola global.

Planificación por grupos: Se planifica un conjunto de hilos afines para su ejecución en un conjunto de procesadores al mismo tiempo.

Asignación dedicada de procesadores: Asignación de los hilos a los procesadores.

Planificación dinámica: El número de hilos puede cambiar en el curso de la ejecución.

Reparto de carga

Ventajas

- Uniformidad de la distribución.
- No es necesario un planificador centralizado.
- La cola puede tener diferentes organizaciones.

Versiones

- FCFS.
- Primero el de menor número de hilos (sin planificar).
- Primero el de menor número de hilos (con expropiación).

Desventajas

- La cola única es un cuello de botella.
- Es improbable que los hilos explusados reinicien su ejecución en el mismo procesador que fue ejecutado. (Perdida del cache).
- Si todos los hilos son tratados como una reserva común, puede que no se logre la sincronización.

Planificación por grupos

Ventajas

- Si los procesos relativamente próximos se ejecutan en paralelo, pueden reducirse los bloqueos por sincronización, menos intercambio de procesos y se incrementa el rendimiento.
- La sobrecarga de planificación puede reducirse debido a que una sola decisión afecta a varios procesadores y procesos al mismo tiempo.

  La planificación por grupos se ha aplicado a la planificación simultánea de hilos que forman parte de un único proceso. Minimiza el intercambio de procesos.

Asignación dedicada de procesadores

Consiste en dedicar un grupo de procesadores a una aplicación mientras dure la aplicación.

1. En un sistema masivamente paralelo, con decenas o cientos de procesadores, la no utilización de uno no decrementa el uso del sistema.
2. La anulación total de intercambio de procesos durante el tiempo de vida de un programa.

Figure 10.2 Example of Scheduling Groups with Four and One Threads [FEIT90]

Planificación en Linux

Clase de prioridad en Linux:

SCHED_FIFO: Hilos de tiempo real con planificación FIFO.

SCHED_RR: Hilos de tiempo real con planificación de turno rotatorio.

SCHED_OTHER: Hilos que no son de tiempo real y otros.

Hilos FIFO

1. El sistema no interrumpe la ejecución de un hilo FIFO, excepto en los siguientes casos:
   a) Pasa a estar Listo otro hilo FIFO de mayor prioridad.
   b) El hilo FIFO en ejecución se bloquea a la espera de un evento, como una E/S.
   c) El hilo FIFO en ejecución abandona el procesador como resultado de la ejecución de la primitiva `sched_yield`.
2. Cuando se interrumpe un hilo FIFO en ejecución, pasa a la cola asociada a su prioridad.

Hilos RR

La política SCHED_RR es similar a la SCHED_FIFO, excepto por el uso de un cuanto de tiempo asociado a cada hilo.

Planificación dinámica

El lenguaje y las herramientas del sistema permite cambiar dinámicamente el número de hilos de un proceso.
Se propone un enfoque en el que tanto el sistema operativo como la aplicación están involucrados en la toma de decisiones de planificación.

- El SO es responsable de repartir los procesadores entre los trabajos.
- Cada trabajo emplea los procesadores de su particiónpara procesar un subconjunto de sus tareas ejecutables, organizando estas tareas en hilos.
- A las aplicaciones individuales se les deja la decisión sobre el subconjunto a ejecutar, además de a qué hilo suspender cuando se expulsa un proceso

1. Si hay procesadores desocupados, se usan para satisfacer la petición.
2. En otro caso, si el trabajo que realiza la petición está recién llegado, se le asigna un procesador individual quitándoselo a algún proceso que tenga más de un procesador asignado.
3. Si no se puede satisfacer alguna parte de la petición, queda pendiente hasta que un procesador pase a estar disponible o hasta que el trabajo anule la petición.
   Al liberar uno o más procesadores:
4. Explorar la cola de peticiones de procesador no satisfechas. Y asignar los procesadores.

A	minimum
B	middle
C	middle
D	maximum

(a) Relative thread priorities

D → B → C → A →

(b) Flow with FIFO scheduling

D → B → C → B → C → A →

(c) Flow with RR scheduling

Figure 10.9 Example of Linux Scheduling

Planificación en UNIX SVR4

El nuevo algoritmo está diseñado para asignar máxima prioridad a los procesos de tiempo real, el siguiente nivel de prioridad a los procesos en modo de núcleo y el nivel más bajo para los proceso en modo de usuario.

1. El uso de planificación por prioridades estáticas preferentes y la introducción de un conjunto de 160 niveles de prioridad divido en tres categorías.
2. La inserción de puntos de apropiación.

Priority Class	Global Value	Scheduling Sequence
Real-time	159 ... 100	first
Kernel	99 ... 60	
Time-shared	59 ... 0	last

Figure 10.10 SVR4 Priority Classes

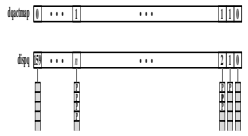


Figure 10.11 SVR4 Dispatch Queues

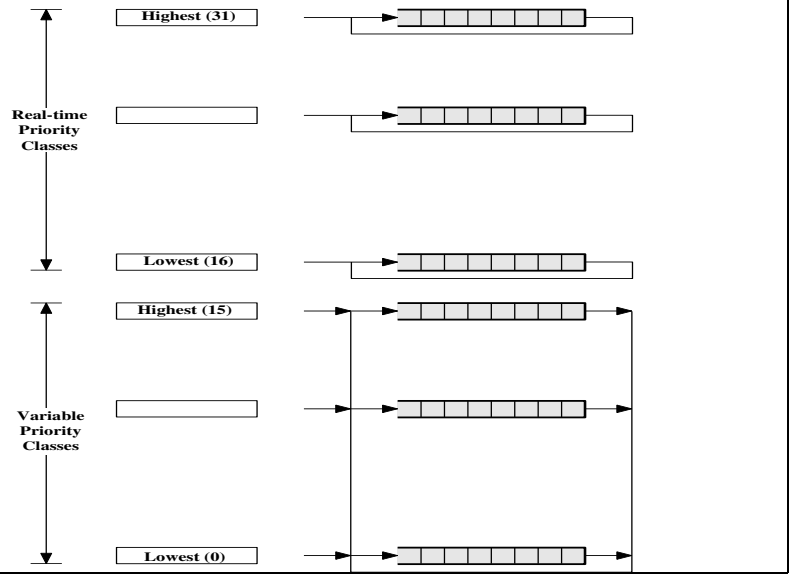


Figure 10.12 Windows 2000 Thread Dispatching Priorities

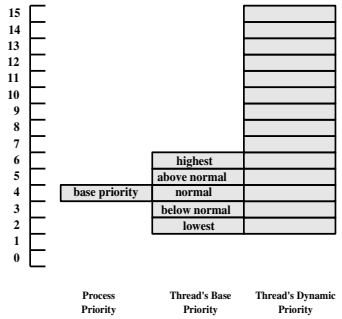


Figure 10.13 Example of Windows 2000 Priority Relationship