

## TEMA 2.- GESTIÓN DE PROCESOS

---

### ÍNDICE:

1. Introducción.....	Pág. 01
2. Concepto de proceso.....	Pág. 04
3. Estados de un proceso.....	Pág. 04
4. Transiciones de estados de los procesos.....	Pág. 05
5. Procesos e hilos (THREADS).....	Pág. 06
6. El bloque de control de procesos.....	Pág. 07
7. Operaciones sobre procesos.....	Pág. 08
8. Interrupciones.....	Pág. 10
9. Criterios de planificación.....	Pág. 15
9.1. Niveles de la planificación.....	Pág. 15
9.2. Objetivos de la planificación.....	Pág. 15
9.3. Criterios de planificación.....	Pág. 16
9.4. Medidas.....	Pág. 16
9.5. Algoritmos de planificación.....	Pág. 17
9.6. Planificación POSIX.....	Pág. 23
9.7. Planificación WINDOWS.....	Pág. 24
10. Mecanismos de comunicación y sincronización.....	Pág. 25

### BIBLIOGRAFÍA:

- [STALLI'01] Capítulos 1,3,4 y 9.
- [CARRET'01] Capítulos 3,11 y 12.
- [MORERA'95] Capítulo 3

## 1.- INTRODUCCIÓN.

A la hora de ejecutar un proceso usamos los registros, los cuales podemos clasificar en dos tipos:

a) Visibles por el usuario: aquellos que el programador utiliza para ejecutar sus aplicaciones. Ejemplos:

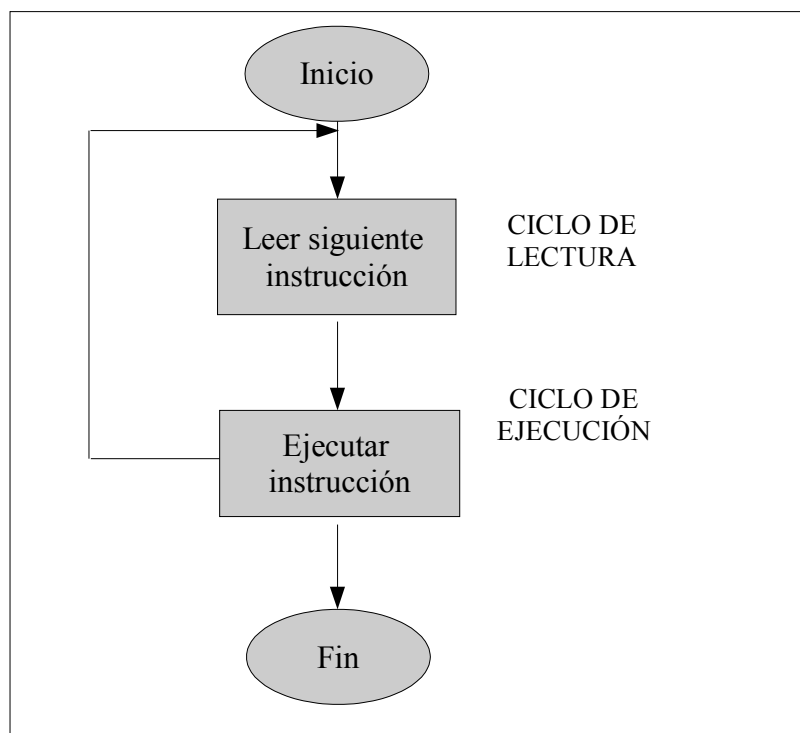
- Registros de datos (AX, BX)
- Registros de dirección, dependen del modo de direccionamiento, el cual puede ser:
  - Indexado (SI, DI)
  - Puntero a segmento (CS,DS)
  - Puntero a pila (SS)
- Registros de condición (Flags), son accesibles por cualquier programa de usuario.

b) Control y estado: no son modificables, lo cual nos hace programar saltos.

- Registros de control y estado.
  - CP (count program)
  - IR (registro de instrucciones) aquí se encuentra la próxima instrucción a ejecutar.
  - PSW (program Status Word) es un registro de flags el cual puede advertirnos cuando existe overflow, si el procesador está en modo núcleo o modo usuario, si determinadas interrupciones están activas o no, etc...

### 1.1.EJECUCIÓN DE INSTRUCCIONES

La ejecución de instrucciones se realiza dependiendo del estado del procesador. El cual siempre está realizando el ciclo de procesado:



Esquema 1. Ciclo de procesado

**1.2. OPERACIONES**

Es decir, las operaciones que puede realizar el procesador al ejecutar un proceso.

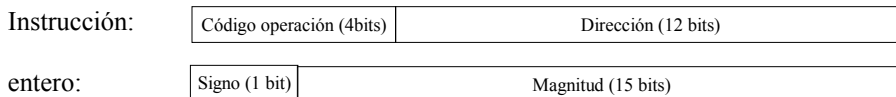
- a) Procesador / Memoria. (transferir información desde el procesador a la memoria, o viceversa)
- b) Procesos E/S. (transferir información de los registros del procesador a los periféricos)
- c) Tratamiento de datos. (aritméticos)
- d) Instrucciones de control. (modificar el valor de PC... )

**1.3. NIVELES DE EJECUCIÓN**

- a) Nivel Usuario: Las direcciones de memoria y ciertas operaciones están limitadas, como acceder a ciertas posiciones de memoria, o ejecutar ciertas instrucciones.
- b) Nivel Supervisor: Podemos hacer cualquier cosa.

Ejercicio 1

Suponemos que tenemos un procesador muy simple, el cual tiene 16 bits de instrucciones y de datos.



Tendremos 3 registros en la CPU:

- PC, contador de programa 16 bits
- IR, Registro de instrucciones 16 bits
- AC, acumulador 16 bits

Conocemos los siguientes códigos de operación:

- 0001 : cargar de la memoria al acumulador.
- 0010 : almacenar el contenido de AC en memoria.
- 0101 : sumar el acumulador y lo de la memoria, el resultado se deja en el acumulador.

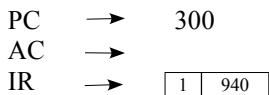
Sabiendo que las posiciones de memoria son:

Memoria			
300	1	940	0
301	5	941	.
302	2	941	.
			.
			.
			.
			.
			.
			.
940		0003	.
941		0002	.

4095      16 ← Si las entradas son de 1 byte

↑  
Si las entradas son de 16 bits

Situación inicial:



¿Cuántos tipos de instrucciones puede tener este microprocesador, es decir, cuál es el juego de instrucciones?

$$2^4$$

¿Qué memoria tiene este equipo?

$$2^{12} \rightarrow 2^2 \cdot 2^{10} = 4 \cdot 1k = 4k = 4096 \text{ posiciones.}$$

Cada una de las posiciones es de 16 bits, es decir, 2 bytes.

Lo que implica que tendremos 8k de memoria.

1<sup>er</sup> ciclo.

PC → 301  
AC → 0003  
IR → 

1	940
---	-----

2<sup>do</sup> ciclo.

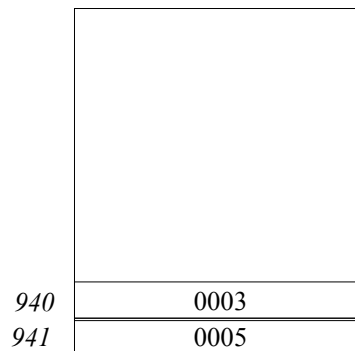
PC → 302  
AC → 0005  
IR → 

5	941
---	-----

3<sup>er</sup> ciclo.

PC → 303  
AC → 0005  
IR → 

2	941
---	-----



Todo esto equivale a la ejecución de una instrucción de alto nivel, en tres ciclos.

$$y = x + y$$

## 2.- CONCEPTO DE PROCESO.

- **Definición informal:** es una instancia del programa en ejecución.
- **Definición formal:** es un programa en ejecución que está formado por el valor actual de: el PC, de los registros y de las variables, siendo además la unidad de procesamiento gestionada por el SO.

De esta manera, un So tiene un montón de procesos que se ejecutan de forma concurrente.

## 3.- ESTADOS DE UN PROCESO.

a) 1º ESTADO. - Inactivo -

Es un pseudo estado. Es un programa que aun no ha sido lanzado, por lo que no llega a llamarse proceso.

b) 2º ESTADO. - Preparado -

Cuando un proceso se crea , es decir, cuando un proceso para ejecutarse, solo necesita que se le conceda el procesador.

c) 3º ESTADO. - En Ejecución -

El proceso que se esta ejecutando en ese instante.

d) 4º ESTADO. - Bloqueado -

Cuando a un proceso le falta el procesador y algún otro recurso, para poder ejecutarse.

**ESTADO GLOBAL DEL SISTEMA.**

Esta formado por el estado de todos los procesos y recursos del sistema en un instante dado.

- Nos basamos en 3 principios básicos:

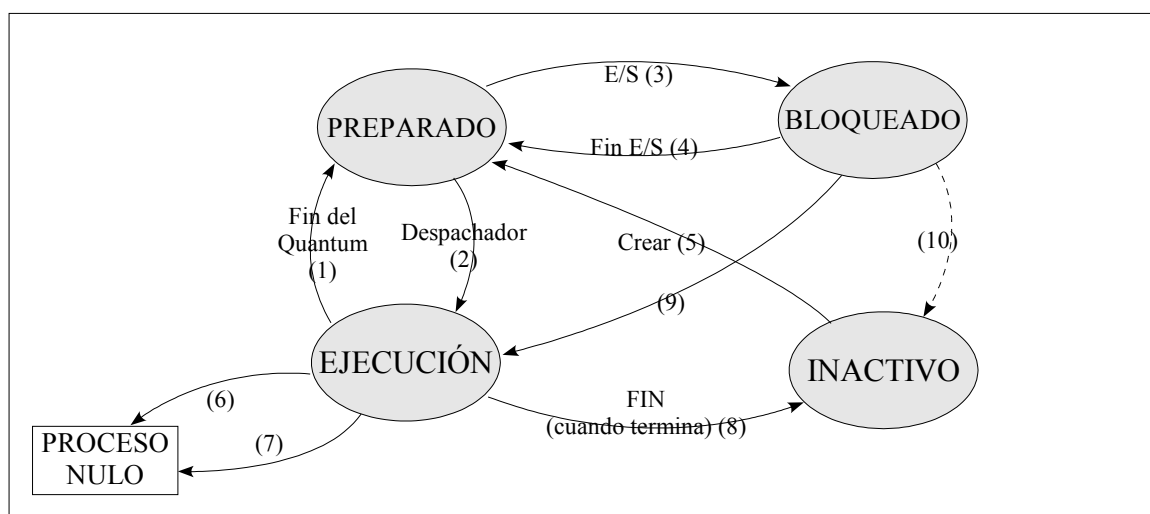
- A .- Hay paralelismo real entre el procesador y la E/S.
- B .- Los procesos tienen fases de alternancia entre el procesador y la E/S.
- C .- Es posible almacenar varios procesos en la memoria.

**4.- TRANSICIONES DE ESTADOS DE LOS PROCESOS.**

El estado global cambia de un instante a otro.

- DIAGRAMA DE TRANSICIÓN DE ESTADOS

Son las situaciones que se pueden dar para que los procesos cambien de estado.



Esquema 2 Diagrama de transición de estados

- (1) Un proceso puede dejar de estar en ejecución si su quantum finaliza.  
Quantum: rodaja de tiempo compartido que se distribuye a los procesos preparados.
- (2) Despachador (distpacher): es el encargado de guardar todos los datos del último proceso ejecutado, y colocar el siguiente en ejecución.  
Planificador: el que determina el orden en el que van a ejecutarse los procesos preparados.
- (3) Al proceso en ejecución, al cual no se le ha acabado el quantum y esta realizando una operación de E/S, se le coloca en bloqueado de forma voluntaria, para ceder el procesador al siguiente que este preparado.
- (4) Cuando el proceso bloqueado termine la entrada salida, se colocará en la lista de preparado.
- (5) Un proceso inactivo, cuando desea hacer uso del procesador pasa a la lista de preparado y espera su turno.
- (6) Si no tenemos ningún proceso para ejecutar. El procesador pasa a ejecutar el 'proceso nulo'.
- (7) Como el proceso nulo no tiene quantum y su prioridad es la mas baja, en cuanto llegue algún proceso al sistema se le quitará el procesador inmediatamente.

(8 ) Podemos encontrarnos con dos situaciones poco habituales:

(9 ) Solo en sistemas de **Tiempo Real**. El proceso pasa de bloqueado a ejecución inmediatamente.

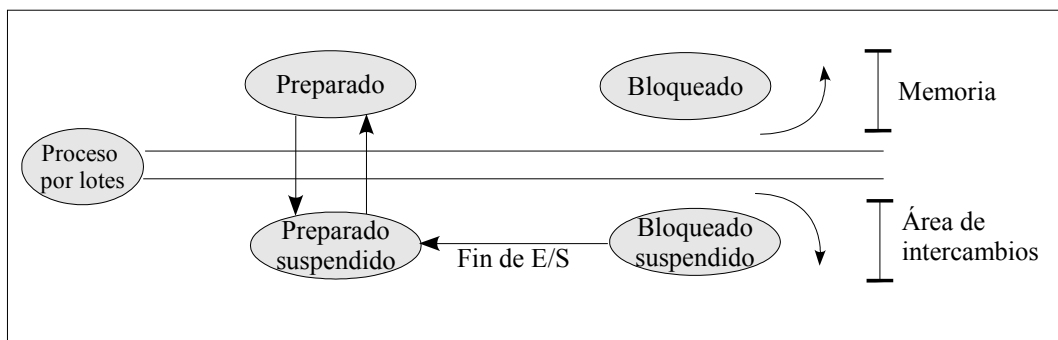
(10 ) Cuando ocurre una **situación de interbloqueo**. El sistema operativo tiene que terminar el proceso inmediatamente.

#### **4.1. ESTADOS SUSPENDIDOS**

Son el reflejo de los estados normales, pero cuando estamos ejecutando un proceso por lotes o para un proceso que este bloqueado.

Estos estados implican un intercambio de ficheros entre la memoria y el área de intercambios.

- a) Preparado suspendido.
- b) Bloqueado suspendido.



Esquema 3 Estados suspendidos

### **5. - PROCESOS E HILOS (THREADS) = proceso ligero, hebra (hilo).**

Las características fundamentales de un proceso son:

- a) Unidad de propiedad de recursos.
- b) Unidad de expedición.

Un proceso puede dividirse en trozos mas pequeños, denominados hilos.

#### **5.1. HILOS**

Programa en ejecución que comparte, la imagen de memoria y otras informaciones, con otros hilos del mismo programa, debido al paralelismo.

- a) Información que tiene un hilo:
  - Estado de ejecución.
  - Contexto del procesador.
  - Pila (de ejecución propia).
  - Almacenamiento de variables locales.
  - Acceso a la memoria y recursos del proceso.
- b) Ventajas respecto a los procesos.
  - Creación y finalización más rápida.
  - Conmutación entre hilos del mismo proceso más fácil.
  - Mejor comunicación entre hilos de un mismo proceso.

## 5.2. PROCESOS.

El proceso, a diferencia del hilo, se queda con la imagen del proceso y con el acceso a los recursos.

- a) Operaciones (ambas implican también la finalización de los hilos)
  - Finalización del procesos.
  - Bloqueo del proceso.
- b) Estados del proceso según el estado de sus hilos:
  - *En Ejecución*, cuando uno de sus hilos se está ejecutando.
  - *Bloqueado*, cuando todos sus hilos están bloqueados.
  - *Preparado*, cuando algún hilo está preparado.

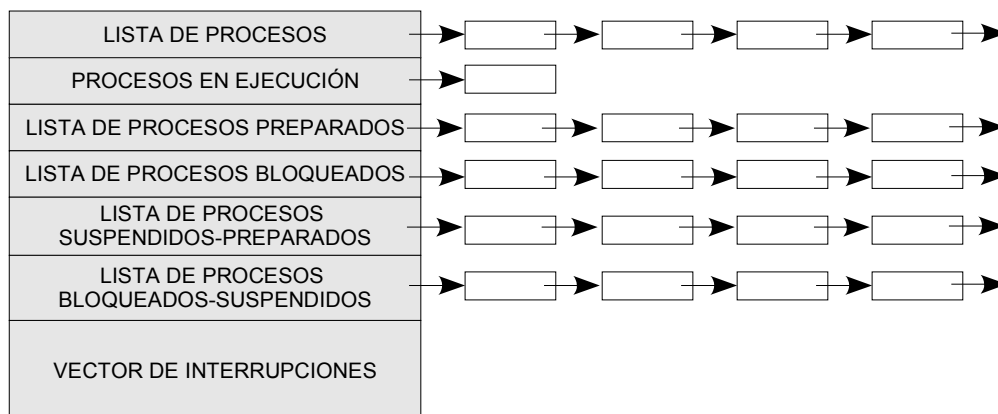
**Los HILOS no pueden estar SUSPENDIDOS.**

## 6.- BLOQUE DE CONTROL DE PROCESOS (PCB).

### 6.1. BLOQUE DE CONTROL DEL SISTEMA (SCB)

Estructura de datos de la que parten todos los PCB.

- a) Contenido del SCB:
  - Lista de procesos, que hay en el sistema.
  - Apuntador a los procesos en ejecución.
  - Apuntador a la lista de procesos que están en ejecución.
  - Apuntador a la lista de procesos bloqueados.
  - Apuntador a la lista de procesos suspendidos-preparados.
  - Apuntador a la lista de procesos bloqueados-suspendidos.
  - Vector de interrupciones, donde se anuncian las direcciones de las rutinas de tratamiento.



Esquema 4 Bloque de Control del Sistema (PSB)

### 6.2. BLOQUE DE CONTROL DE PROCESOS (PCB)

Estructura de datos, normalmente de tamaño fijo, que define a un proceso.

- a) Contenido de PCB:
  - Identificador del proceso.
  - Derechos (privilegios).
  - Prioridad (es numérica, y depende de la especificación previa).
  - Estado de HW (ó Pc, estado del programa).
  - Estado.

- Recursos (que necesita).
- Proceso Padre (el proceso que lo creó)(inix, en Linux).
- Procesos Hijos (proceso que ha creado).

b) Ventajas:

- Eficiencia.
- Posibilidad de compartir información.

c) Objetivos:

- Poder localizar la información de un proceso.
- Preservar la información de dicho proceso (tener un lugar donde guardar los datos de proceso cuando este no esté en ejecución).

## **7.- OPERACIONES SOBRE PROCESOS.**

### **7.1. CREAR.**

Cuando creamos un proceso debemos realizar las siguientes:

a) Operaciones:

- Asignar un identificador al proceso (PID).
- Obtener un PCB (reservar o preparar un espacio en memoria para él).
- Rellenar la información del procesos.
- Asignar los recursos (los que sean asignables).
- Colocar en la lista de procesos preparados.

b) Clasificación:

- *Creación Jerárquica.*

Para que un proceso exista debe ser creado, previamente, por otros. Por lo que todo proceso tiene un antecesor y puede tener varios predecesores.

- *Creación no Jerárquica.*

Los procesos se crean sin relaciones padre e hijo (windows NT).

### **7.2. DESTRUIR.**

a) Operaciones:

- Liberar los recursos utilizados por el proceso.
- Liberar el espacio que se reservó para el PCB.

b) Operaciones para:

- Creación Jerárquica.

Si procedemos de la forma anterior, se nos ropería el árbol. Por lo que, cuando eliminemos un proceso tendremos que:

- Eliminar a su vez todos los procesos relacionados con él.

o

- Pasar los hijos del proceso a eliminar, al padre.

(en Unix, los hijos no se terminan, por lo que se quedan huérfanos).

- Creación no Jerárquica.  
No habrá problemas.

**7.3. CAMBIAR LA PRIORIDAD.**

**7.4. DORMIR O RETARDAR.**

Subestado de bloqueo, es decir, es un bloqueo que dura un tiempo predeterminado.

**7.5. DESPERTAR.**

**7.6. DESPACHAR.**

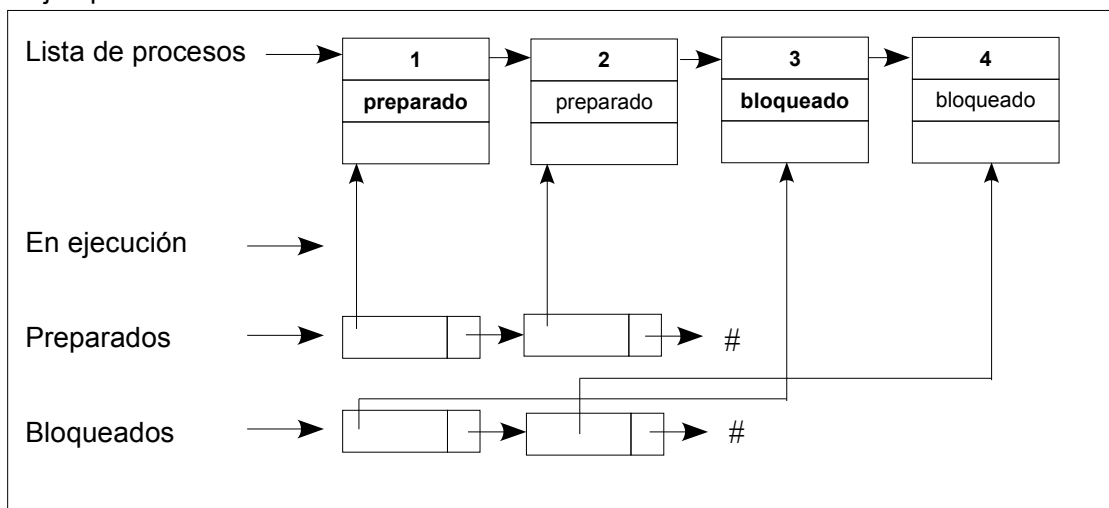
**7.7. ABORTAR.**

Terminar un programa de forma anormal.

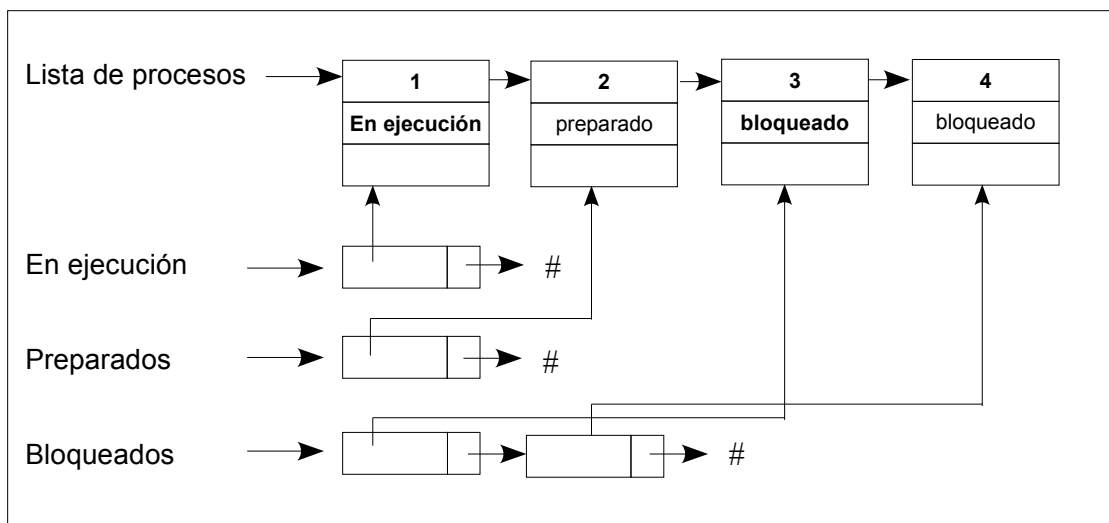
**7.8. SUSPENDER.**

**7.9. DESBLOQUEAR.**

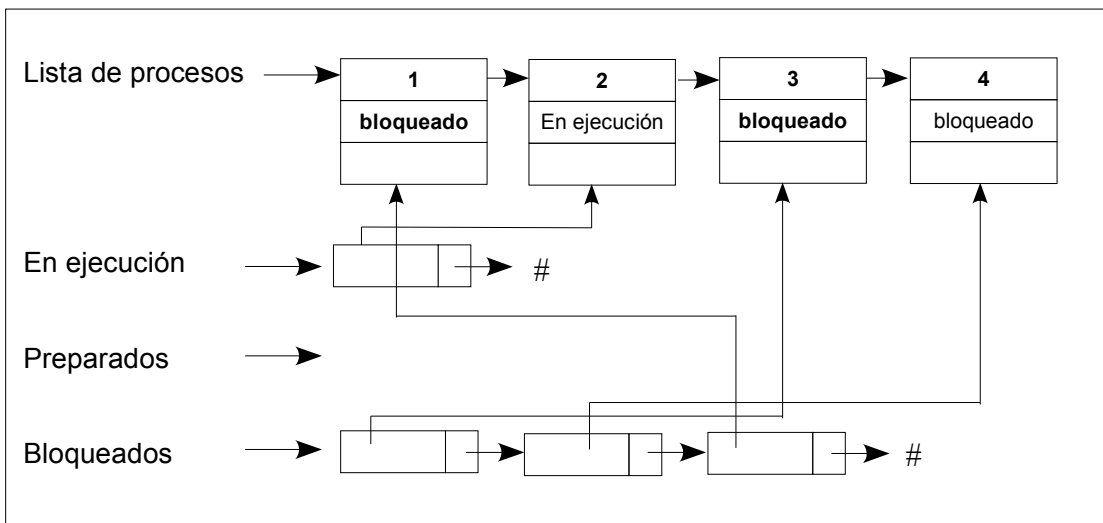
REPRESENTACIÓN DE LOS PROCESOS SEGÚN SU ESTRUCTURA DE DATOS:  
Ejemplos



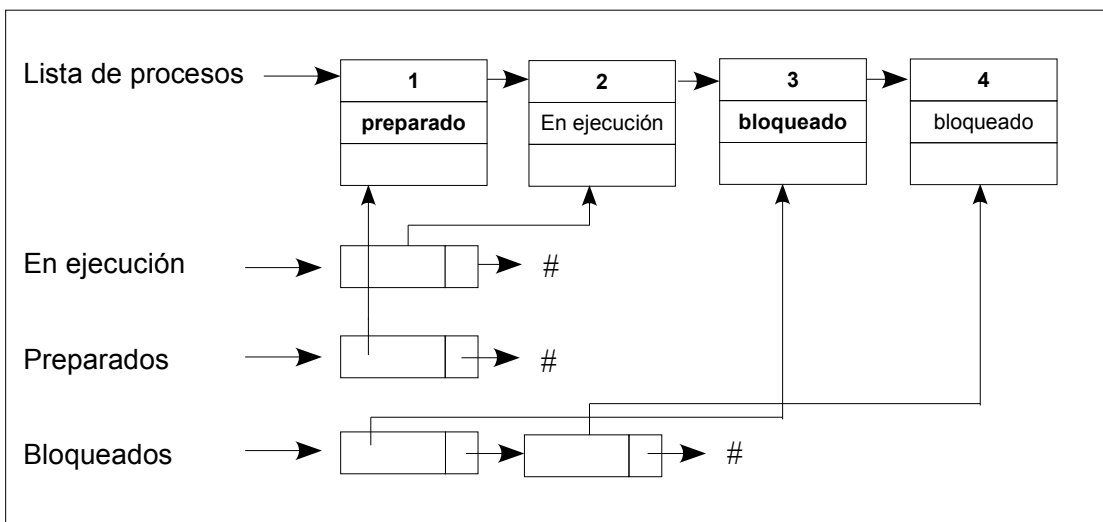
EL PROCESO 1 PASA A EJECUCIÓN



PETICIÓN DE LECTURA DEL DISCO DEL PROCESO 1 (por lo que pasa a bloqueados)



EL PROCESO 1 RECIBE LA INFORMACIÓN QUE SOLICITÓ (pasa a preparado)



**8. - INTERRUPCIONES.**

Es la presencia de un evento o señal que obliga al SO a tomar el control del procesador para estudiarla y tratarla.

**8.1. UTILIDADES DE LAS INTERRUPCIONES.**

- a) El SO toma el control ante:
  - Un error del hardware.
  - Un error del proceso que esta en ejecución.
  - La aparición de una excepción.

- b) Simultanear el uso del procesador y de la E/S



El control de E/S se puede hacer mediante:

- Una interrupción a su término.
- Un chequeo continuo del estado del dispositivo.

c) Reparto de tiempo (tiempo compartido)

Para realizar un reparto de tiempo necesitamos necesariamente las interrupciones del reloj. De modo que cada vez que llega una interrupción del reloj le quitamos una cantidad determinada al quantum del proceso.

d) Reconocer eventos externos

Es decir, la forma en la que los dispositivos nos avisan

### **8.2. TIPOS DE INTERRUPCIONES.**

a) Hardware

- Son interrupciones asíncronas (llegan sin avisar).
- Están generadas por algún periférico.
- Las más comunes son:
  - Interrupciones de E/S a los dispositivos.
  - Interrupciones del procesador.

b) Software

- Son generadas por el proceso en ejecución (llamadas al sistema. Son interrupciones porque hacen que el SO tome el control y ejecute una instrucción)

c) Excepciones

- Son errores de Hardware y Software que generan situaciones inconsistentes (anómalas)
- Pueden ser:
  - Fallos hw y sw.
    - Ejemplo 1: Se desconecta el IDE del CD-ROM.
    - Ejemplo 2: Un programa con división por cero.
  - Eventos anómalos:
    - Se desborda una variable de coma flotante.
  - Datos de entrada incorrectos por parte del usuario.
    - Controlado por los programas.

### **8.3. TIPOS DE FALLOS.**

- a) Catastróficos.
- b) No recuperables.
- c) Recuperables.

### **8.4. EJEMPLO DE EJECUCIÓN DE UN PROCESO CON Y SIN INTERRUPCIÓN**

1,2,3,4,5 = Conjuntos de instrucciones.

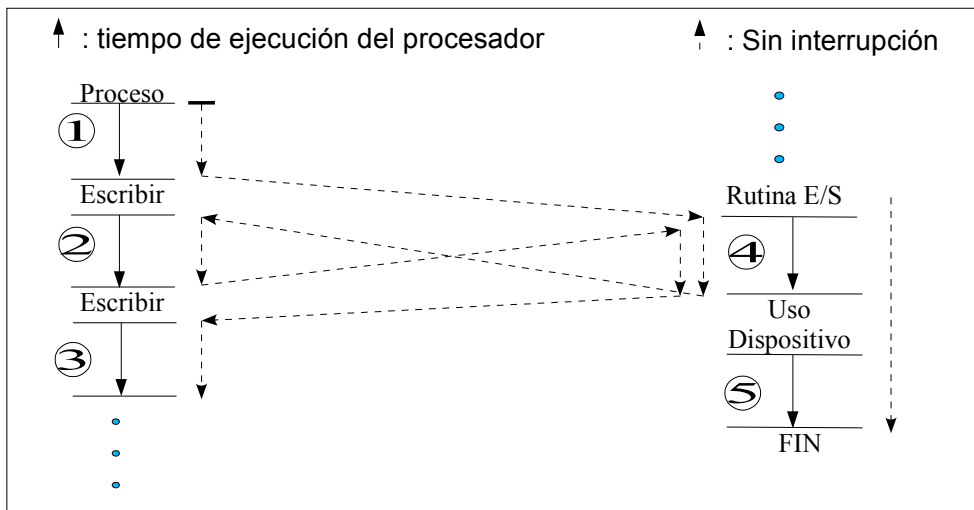
Escribir → Operaciones de E/S

\* : Interrupción

a) Sin interrupción.

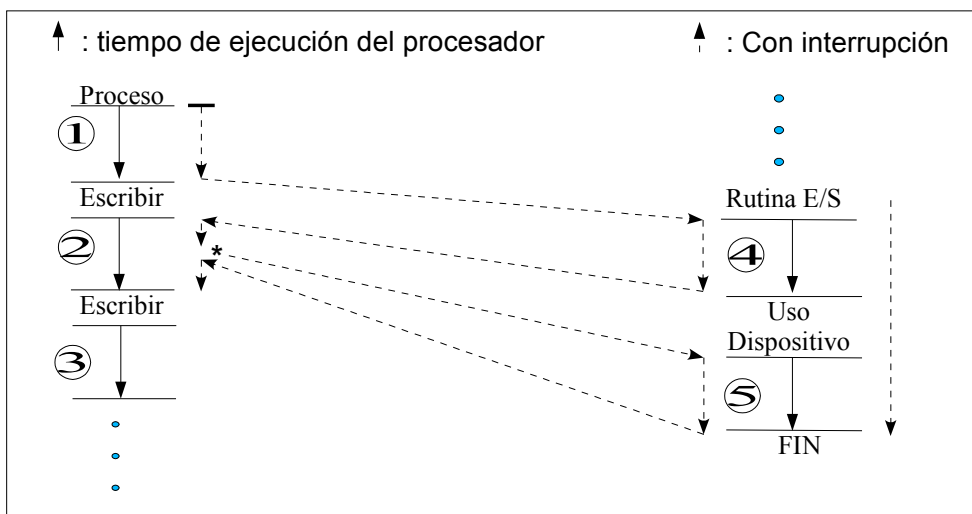
- Para atender la orden de E/S se tiene que preparar una rutina de E/S (esta prepara al dispositivo para mandar los datos, por lo que es necesario un tiempo de procesamiento).
- Usa el dispositivo, y cuando termina debe recoger los datos guardados en el sitio correcto.

Tiempo usado: 1 + 4 + (tiempo indeterminado) + 5 + 2 + 4 + (tiempo indeterminado) + 5 + 3



Esquema 5. Ejecución de un proceso SIN interrupción

b) Con Interrupción.



Esquema 6. Ejecución de un proceso CON interrupción

Tiempo usado: 1 + 4 + 2 a + 5 + 2 b + 4 + 3 a  
 Hemos ganado tiempo.

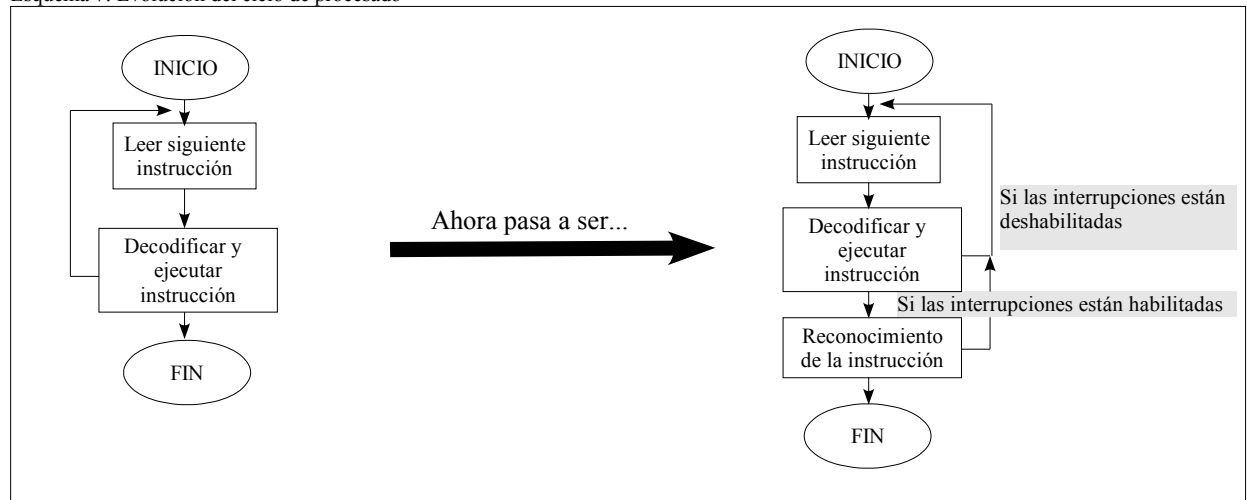
**8.5. PASOS PARA LA GESTIÓN DE INTERRUPCIONES.**

La llegada de una interrupción se atiende según la rutina de interrupción de esta.

- PASO 1: El dispositivo da la señal de interrupción.
- PASO 2: El procesador finaliza la instrucción en curso.
- PASO 3: El procesador comprueba la asistencia de la interrupción y envía señales de reconocimiento.

- PASO 4: Se guarda el valor de PC y del PSW en la pila.
- PASO 5: Se carga en el PC la dirección de la rutina de tratamiento de interrupción.

Esquema 7. Evolución del ciclo de procesado



### - RSI (Rutina de Servicio de Interrupción)

Sus objetivos son:

- Salvar el resto de los valores del registro (en la pila).
- Atender la interrupción.
- Restaurar desde la pila todo los valores menos el PC y el PSW
- Devolver los valores de PC y PSW (cambiando al mismo tiempo el modo de trabajo, es decir, modifica PSW).

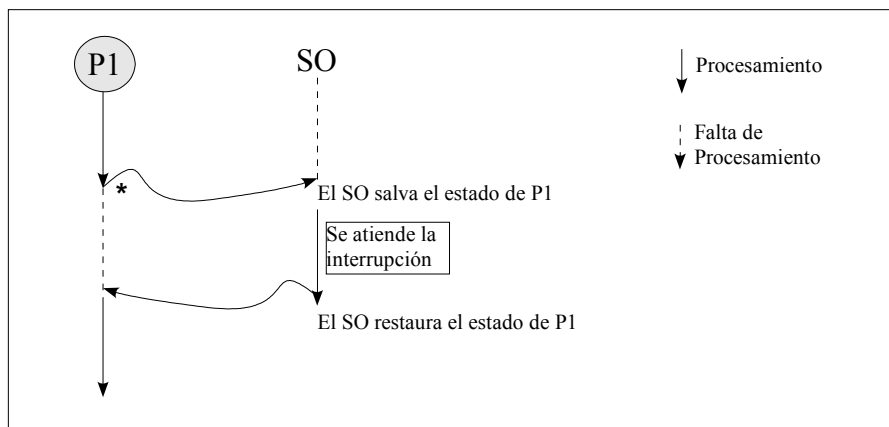
### 8.6.DIFERENCIAS ENTRE INTERRUPCIÓN Y EXCEPCIÓN.

- Las interrupciones :
  - Están relacionadas con el proceso en ejecución.
  - Afectan al proceso y a su entorno
  - Se definen en niveles de prioridad.
  - Dentro de una interrupción puede darse otra interrupción
- Las excepciones:
  - No están relacionadas con los procesos en ejecución.
  - Afectan a todo el sistema.
  - No se define en niveles de prioridad.

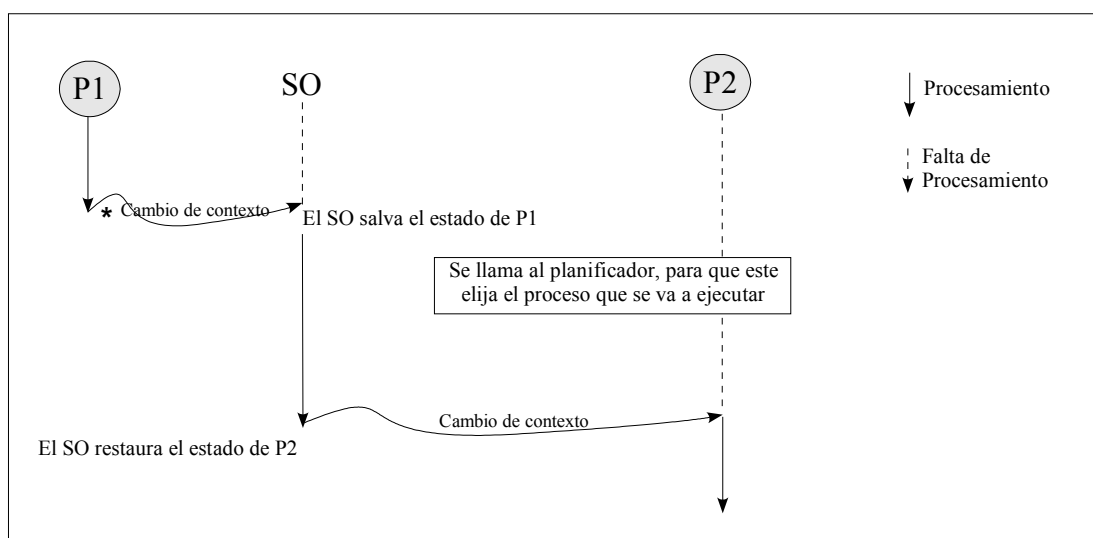
### 8.7. CAMBIO DE CONTEXTO.

Cambio de contexto: Ocurre cada vez que se cambia del modo de trabajo del procesador. Los diferentes modos de trabajo son:

- Supervisor.
- Usuario.
- La atención de una interrupción conlleva dos cambios de contexto.  
Ejemplos:



Esquema 8. Ejemplo de cambio de contexto



Esquema 9. Ejemplo de cambio de un proceso a otro. Implica 2 cambios de contexto y la intervención del SO.

### **8.8. INTERRUPTIONES MÚLTIPLES.**

Las alternativas que tenemos cuando, durante el tratamiento de una interrupción, llega una interrupción nueva, son:

a) Deshabilitar las interrupciones:

Activamos la señal de no reconocimiento de interrupciones (estando en modo supervisor) mientras estamos atendiendo la interrupción.

- Ventaja:  
Las interrupciones se atienden de forma secuencial.
- Desventaja:  
No se puede establecer prioridad entre las interrupciones.

b) Establecer prioridades entre interrupciones:

Cuando llegue una interrupción durante la ejecución de otra, compararemos sus prioridades.

- Si es inferior, se desecha.
- Si es superior, se sustituye por la que este en ejecución.

**Núcleo del SO**, es el encargado de gestionar las interrupciones.

- Funciones del núcleo:
  - A Atender y tratar interrupciones (manejo de las int).
  - B Creación y destrucción de procesos.
  - C Cambiar el estado de los procesos.

- D Despachar los procesos.
- E Dormir y despertar los procesos.
- F Comunicar y sincronizar los procesos entre si.
- G Manipular los PCB
- H Dar soporte al resto de funciones del SO, como son:
  - Gestión de memorias.
  - Gestión y administración del sistema de archivos.
  - Sistema de E/S.

## **9. - PLANIFICACIÓN DEL PROCESADOR**

La planificación consiste en elegir el orden en el que atenderemos a los procesos que se van a ejecutar. Existen tres niveles de planificación.

### **9.1. NIVELES DE PLANIFICACIÓN.**

#### **a) Planificador a largo plazo.**

- Decide que procesos pasan de inactivos a preparados o viceversa.
- Los procesos no se cargan en memoria hasta que el sistema esté preparado.
- Algunos de estos procesos son:
  - procesos interactivos.
  - Procesos por lotes.
- Periodo de ejecución: puede ser de minutos, ya que estos procesos se lanzan muy de vez en cuando y se chequea el grado de multiprogramación del sistema.

#### **b) Planificador a medio plazo.**

- Decide que procesos pasan de bloqueado a suspendido o viceversa.
- Solo existe en sistemas que tienen memoria Swap (intercambio)
- Periodo de ejecución: es de segundos. En este caso decide el grado de multiprogramación del sistema.

#### **c) Planificador a corto plazo.**

- Decide que procesos pasan de preparados a ejecución.
- Es el que más nos interesa.

### **9.2. OBJETIVOS DE PLANIFICACIÓN.**

#### **a) La justicia:**

Intentamos que los procesos no salieran ni beneficiados ni perjudicados.

#### **b) Máxima capacidad de ejecución:**

Intentamos disminuir los cambios de contexto.

Cuanto antes termine un proceso y menos interrupciones haya, menos cambios de contexto habrá.

#### **c) Máxima capacidad de usuarios interactivos:**

Intentamos dar servicio a todos los usuarios conectados.

#### **d) Minimizar la sobrecarga:**

Intentamos que se gaste la menos cantidad de tiempo posible en la administración del sistema.

#### **e) Equilibrio en el uso de recursos: (muy difícil de conseguir).**

Intentamos que la utilización de recursos sea equilibrada.

#### **f) Garantizar el cumplimiento de las prioridades:**

Intentamos que los procesos de tiempo real se ejecuten antes que los procesos interactivos.

Estos objetivos son contradictorios unos con otros, por lo que tendremos que ir sacrificando unos para mejorar otros.

### **9.3. CRITERIOS DE PLANIFICACIÓN.**

**a) Tiempo de espera:**

Es el tiempo que pasa desde que un usuario lanza un trabajo hasta obtiene una respuesta.

- No se mide, porque depende del usuario (subjetivo).

**b) Tiempo de servicio:(Linux)**

Es el tiempo que pasa desde que se lanza la tarea hasta que finaliza su ejecución.

- $T.\text{servicio} = T.\text{carga} + T.\text{espera} + T.\text{procesador} + T.\text{de.E/S}$

**c) Tiempo de ejecución:**

Es el tiempo teórico que necesita un proceso para ejecutarse como si fuera el único que estuviera en el sistema (o estuviera en un SO monotarea)

- Se tiene en cuenta el tiempo de E/S
- $T.\text{Ejecución} = T.\text{procesador} + T.\text{de.E/S}$

**d) Tiempo de espera:**

Es el tiempo en el que el proceso no está usando el procesador, es decir, está bloqueado o preparado.

**e) Tiempo de procesador:**

Es el tiempo en el que se está usando el procesador.

**f) Eficiencia:**

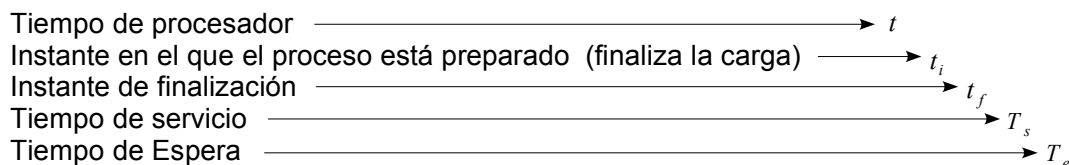
- Es una medida global de sistema.
- Relaciona el tiempo de espera de cada proceso y la ocupación total del procesador.

$$\text{Eficiencia} = \frac{\text{Tiempo de espera}}{\text{Tiempo total de procesador}}$$

**a) Rendimiento:**

Es el número de procesos ejecutados por unidad de tiempo.

### **9.4.MEDIDAS**



a) TIEMPO DE SERVICIO:  $T_s = t_f - t_i$

b) TIEMPO DE ESPERA:  $T_e = T_s - t$

c) INDICE DE SERVICIO:  $I = \frac{t}{T_s} \quad 0 \leq I \leq 1$

- El proceso está “limitado por proceso”, cuando tiende a 1.
- El proceso está “limitado por E/S”, cuando tiende a 0.

b) TIEMPO MEDIO DE SERVICIO:  $T_{ms} = \frac{\sum_{i=1}^n T_s}{n}$

c) TIEMPO MEDIO DE ESPERA: 
$$T_{me} = \frac{\sum_{i=1}^n T_e}{n}$$

d) INDICE MEDIO DE SERVICIO: 
$$I_m = \frac{\sum_{i=1}^n I}{n}$$

### 9.5. ALGORÍTMOS DE PLANIFICACIÓN

Evaluamos la forma de mejora que se aprecia en el sistema con diferentes planificaciones. Como medida para esta evaluación usaremos los tiempos medios, y una unidad de referencia que será la carga de referencia.

#### Características:

- T.Ejecución = T.procesador = t (cuando no haya E/S).
- La unidad de tiempo es adimensional.

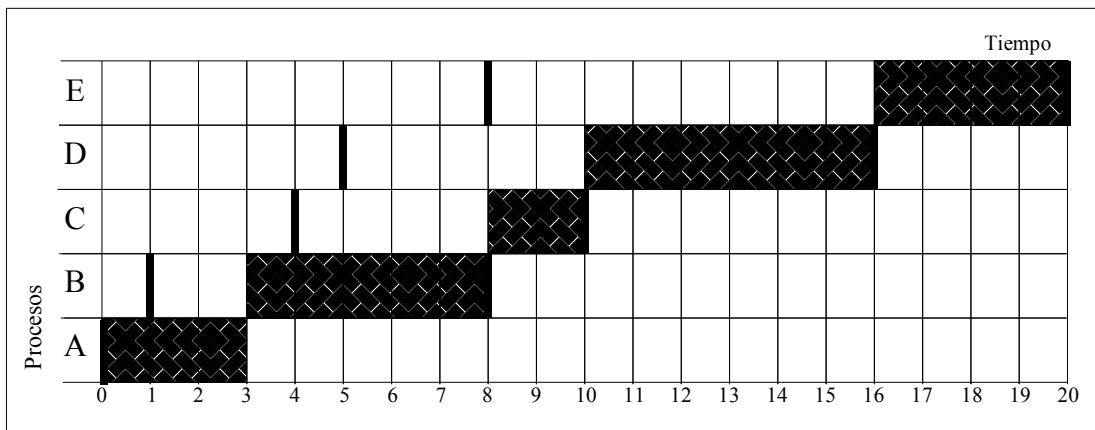
#### Tipos de políticas de planificación:

- Apropiativas: Si se le puede quitar el procesador al proceso que esté ejecutándose, en el caso que haya otro proceso a ejecutar con mejores características.
- No Apropiativas: El proceso no abandona el procesador una vez que haya comenzado su ejecución.

#### **b) POLÍTICA FIFO - FCFS (first coming first service).**

- Su forma de actuar es la de una cola
- Es no apropiativa
- Se basa en la llegada de los procesos.
- Los procesos cortos se ven perjudicados.
- El cálculo de  $T_s$ ,  $T_e$ , e  $I$ , lo hacemos mediante el diagrama de ocupación del procesador.

	$t_i$	t	$t_f$	$T_s$	$T_e$	I
A	0	3	3	3	0	$3/3=1$
B	1	5	8	7	2	$5/7 = 0.71$
C	4	2	10	6	4	$2/6 = 0.33$
D	5	6	16	11	5	$6/11 = 0.54$
E	8	4	20	12	8	0.33
				7'8	3'8	0'58
				$T_{ms}$	$T_{me}$	$I_m$



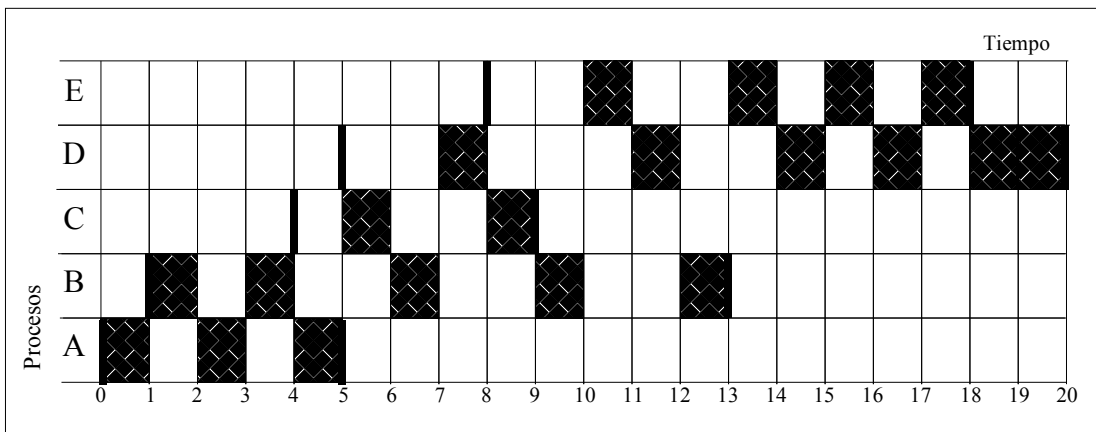
Esquema 10. Diagrama de ocupación del procesador – FIFO -

- CARACTERÍSTICAS:
  - Es Justa.
  - Es predecible el tiempo total de ejecución.
  - $I_m$  depende del número de procesos que haya en la cola.

**c) POLÍTICA ROUND ROBIN**

- Para sistemas que usen el tiempo compartido.
- Funcionamiento:
  - Intenta repartir el tiempo de ejecución entre los procesos de la cola:
  - Se le concede, a cada proceso de la lista de preparados, un Quantum (la cantidad de tiempo que usará el procesador).
  - La lista de preparados funcionará como una fifo.
- Con QUANTUM = 1

	$t_i$	$t$	$t_f$	$T_s$	$T_e$	$I$
A	0	3	5	5	2	$3/5=0'6$
B	1	5	13	12	7	$5/12 = 0'42$
C	4	2	9	5	3	$2/5 = 0'4$
D	5	6	18	15	9	$6/15 = 0'4$
E	8	4	20	10	6	$4/10 = 0'4$
				<b>9'4</b>	<b>5'4</b>	<b>0'44</b>
				$T_{ms}$	$T_{me}$	$I_m$



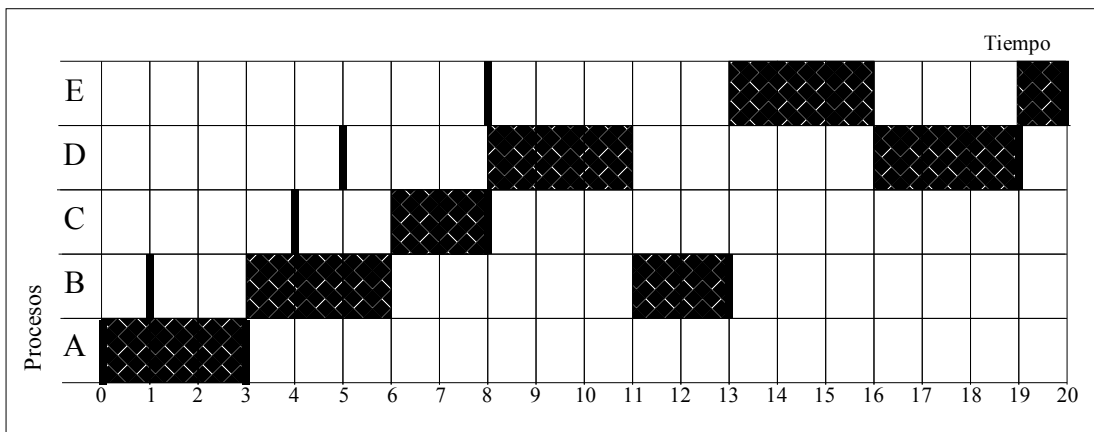
Esquema 11. Diagrama de ocupación del procesador ROUND-ROBIN (Q=1)

Hay demasiados cambios de contexto, por lo que perdemos mucho rendimiento ya que aumentamos la carga del sistema.

Si un proceso finaliza y aún le queda Quantum continua el tiempo en ese proceso hasta que se termine.

- Con QUANTUM = 3

	$t_i$	$t$	$t_f$	$T_s$	$T_e$	$I$
A	0	3	5	5	0	1
B	1	5	13	12	7	0'42
C	4	2	8	4	2	0'5
D	5	6	19	14	8	0'43
E	8	4	20	12	8	0'33
				9	5	0'54
				$T_{ms}$	$T_{me}$	$I_m$



Esquema 12. Diagrama de ocupación del procesador ROUND-ROBIN (Q=3)

• Características:

- Muy utilizada en el tiempo compartido.
- $I_s$  es uniforme para todos los procesos.
- La sobrecarga del sistema depende de la eficiencia de los cambios de contexto (aún así la sobrecarga es muy leve).

**d) POLÍTICA SJF (Short Job First).**

• Funcionamiento:

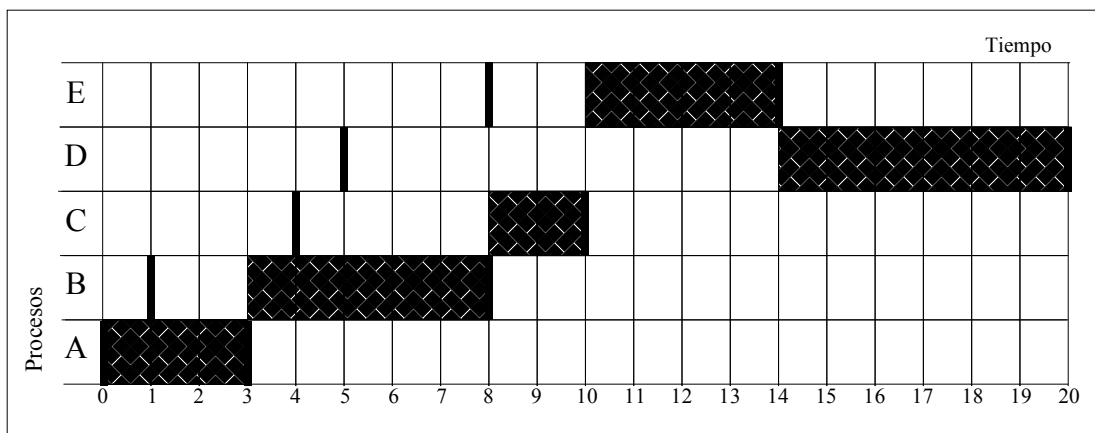
Elegir de la lista de preparados al proceso cuyo tiempo de ejecución sea más corto.

- En esta política sí interviene el Tiempo de E/S.
- El conocimiento previo del tiempo de ejecución de cada proceso puede darse por:
  - Que el usuario lo indique.
  - Que el proceso se auto describa. (lo debe calcular el compilador, pero da problemas ya que los procesos no siempre se comportan de la misma manera)
  - La experiencia previa (heurística). No es normal que un proceso tenga una experiencia previa en el sistema.

• Características:

- Poco predecible. (puede tener postergación indefinida).
- No es justa. (favorece a los procesos cortos).
- Índices de servicio buenos, pero es muy difícil llevar la política a la práctica.

	$t_i$	$t$	$t_r$	$T_s$	$T_e$	$I$
A	0	3	3	3	0	1
B	1	5	8	7	2	0'71
C	4	2	10	6	4	0'33
D	5	6	20	15	9	0'4
E	8	4	14	6	2	0'67
				3'4	7'4	0'62
				$T_{ms}$	$T_{me}$	$I_m$

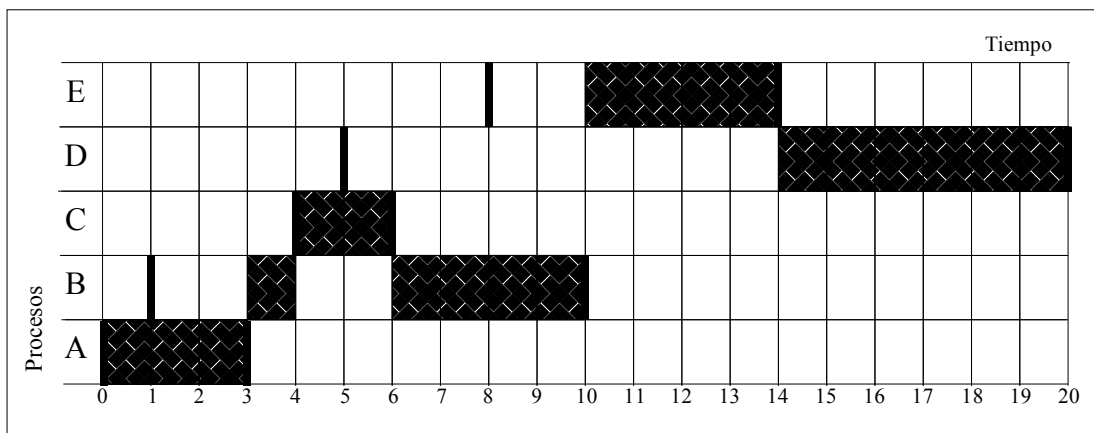


Esquema 13. Diagrama de ocupación del procesador – SJF -

**e) POLÍTICA SRT (Short Remaining Time)**

- Funcionamiento:
  - Se escoge de la lista de preparados el proceso más corto.
    - Apropiativa, es decir, si llega un proceso más corto que el que se este ejecutando, este último pasa a ejecución.
- Características:
  - El Índice de servicio muy bueno.
  - El Tiempo de espera es pequeño para la mayoría de los procesos.
  - Es justa.
  - Produce bastante sobrecarga, por culpa de los cálculos necesarios para saber que proceso es el más corto.

	$t_i$	$t$	$t_r$	$T_s$	$T_e$	$I$
A	0	3	3	3	0	1
B	1	5	10	9	4	0'55
C	4	2	6	2	0	1
D	5	6	20	15	3	0'4
E	8	4	14	6	2	0'67
				7	3	0'72
				$T_{ms}$	$T_{me}$	$I_m$



Esquema 14. Diagrama de ocupación del procesador - SRT -

**f) POLÍTICA HRN (High Response Next).**

- Funcionamiento:
  - Calcula a cada instante el tiempo relevante de cada proceso.

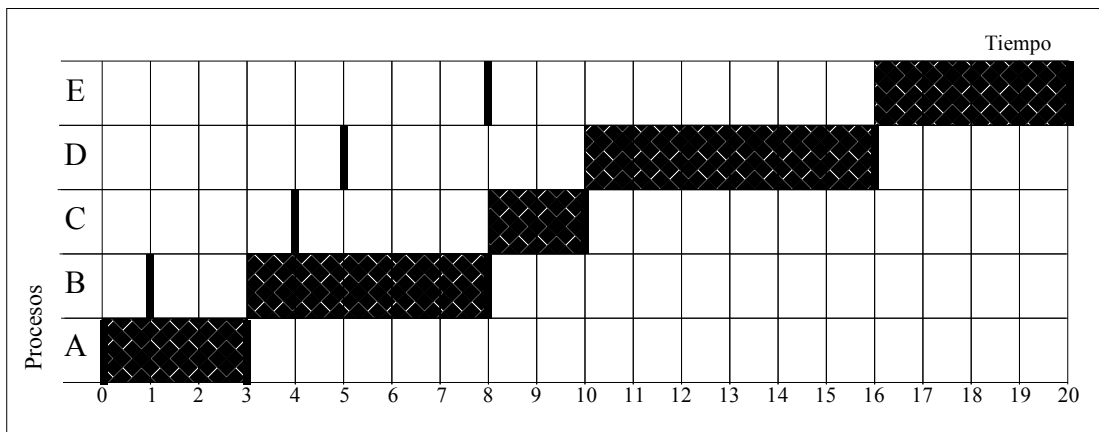
- Tiempo relevante = 
$$p = \frac{(w+t)}{t}$$

$t$ : tiempo de ejecución.  
 $w$ : tiempo que lleva el proceso en la lista de preparados.

Se ejecuta el que tenga  $p$  más grande. Todos los procesos empiezan por  $p=1$ .

- Características:
  - Justa.
  - Tiene mucha sobrecarga. Debido al cálculo de  $p$ , el cual requiere una operación de suma y otra de división, del cual se tiene que ir llevando la cuenta.

	$t_i$	$t$	$t_f$	$T_s$	$T_e$	$I$
A	0	3	3	3	0	1
B	1	5	8	7	2	0'71
C	4	2	10	6	4	0'33
D	5	6	16	11	5	0'54
E	8	4	20	12	8	0'33
				$T_{ms}$	$T_{me}$	$I_m$
				7'8	3'8	0'58



Esquema 15. Diagrama de ocupación del procesador - HRN -

Curiosamente nos a dado igual que una fifo, por lo que no nos compensa respecto a la fifo, debido a la sobrecarga.

### g) POLÍTICA DE COLAS DE MÚLTIPLES HILOS (MQ)

- Funcionamiento:
  - Cómo los procesos tienen distinta naturaleza, procedemos a dividir la lista de procesos preparados, en varias listas con diferentes tipos de planificación.
  - Según van llegando los procesos se les van asignando de forma permanente a una lista.
  - Mediante una política de prioridades entre colas, decidimos qué proceso se ejecutará.
  - Se usan los procesos de la cola menos prioritaria, cuando no queden en la más prioritaria.
  - Apropiación entre colas.

### h) POLÍTICA FB (First Back), realimentación de colas múltiples.

- Funcionamiento:
  - Nuestro objetivo es que se ejecuten primero los procesos cortos y luego los largos, para ello:
    - 1º) Establecemos un conjunto de colas, donde los procesos pueden cambiar de cola según el tiempo que lleven en el sistema.
    - 2º) Cada cola tendrá una política de planificación.
    - 3º) Los procesos pueden comenzar en distintas colas.
    - 4º) Existe apropiatividad entre colas. Las superiores son las más prioritarias.
 A cada proceso se le asigna un tiempo en una cola, si se le acaba el tiempo pasa al nivel inferior.
  - Necesitamos conocer:
    - Número de colas.
    - Planificación de cada cola.
    - Formas de pasar de una cola a otra.

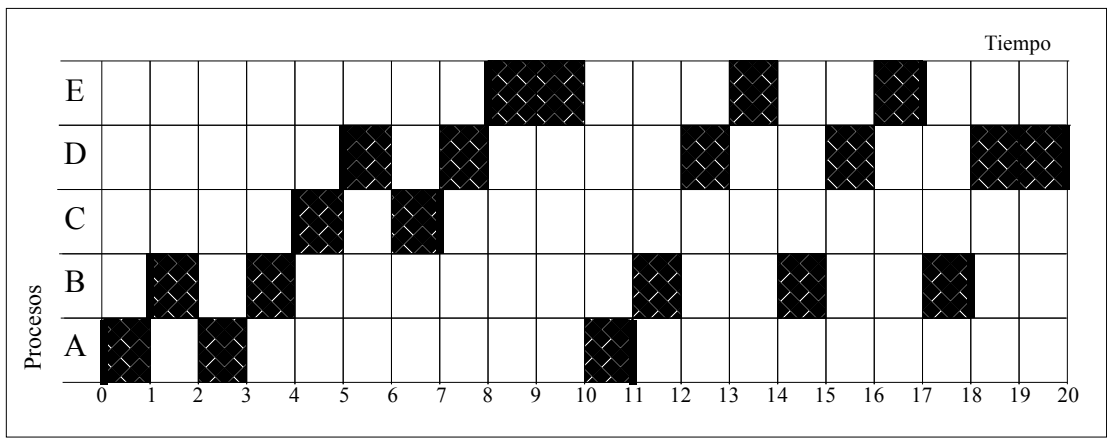
- Cola inicial de cada proceso.
- Política de planificación entre colas.

• Características:

- Es apropiativa entre colas.
- Soporta bien una carga de procesos alta.
- Se adapta a las necesidades del sistema.

	$t_i$	$t$	$t_f$	$T_s$	$T_e$	$I$
A	0	3	11	11	8	0,27
B	1	5	18	17	12	0'29
C	4	2	7	3	1	0'67
D	5	6	20	15	9	0'4
E	8	4	17	9	5	0'44
				7'8	3'8	0'41
				$T_{ms}$	$T_{me}$	$I_m$

En este caso:  
 · N° colas = 6  
 · Round Robin de  $q = 1$   
 · Por el final  
 · Todos en la 1ª  
 · Apropiativa con prioridades.  
 1ª + prioridad  
 6ª - prioridad.



Esquema 16. Diagrama de ocupación del procesador - FB -

**i) POLÍTICA DE PRIORIDADES DINÁMICAS.**

- Funcionamiento:
  - Se establece una cola por prioridad. Pero podemos encontrarnos con postergación indefinida.
  - Para evitar la postergación indefinida, usamos las prioridades dinámicas (que puedan cambiar durante la ejecución). Cómo hacer que la prioridad aumente según el tiempo que lleve el proceso en esperando.

**9.6.PLANIFICACIÓN POSIX.**

Linux se basa en el estándar Posix, el cual planifica el sistema basándose en:

a) La prioridad.

(Establece que debe haber 32 niveles).

b) La política de planificación.

- La planificación seleccionará el proceso o hilo con prioridad mayor. Dicho proceso o hilo que sea seleccionado se ejecutará según la política de planificación que tenga asociado.
- Las políticas de planificación que define Posix son:

**A FIFO.**

Funcionamiento:

- Como una FIFO normal, pero atendiendo a las siguientes reglas:
  - Si un proceso es expulsado por otro de mayor prioridad, se le coloca el primero en la fila de su prioridad.
  - Si se bloquea un proceso, al volver se coloca el último de su cola.
  - Si se cambia la prioridad o la política y a causa de esto es expulsado el proceso, pasará a estar el último de su cola.

**B Round Robin**

Funcionamiento:

- Como una Round Robin pero atendiendo a las siguientes reglas:
  - Cuando finaliza el quantum pasa al final de su cola.
  - Cuando llega un proceso de mayor prioridad mitad de un quantum, es expulsado y pasará como el primero de su cola, con el quantum que le quedaba.

c) LINUX.

- Prioridad.

Define 100 niveles de prioridad de las cuales:

- *Las que sean distintas a 0*, serán estáticas y de tiempo real. Para estas prioridades se implementan FIFO y Round Robin según Posix.
- *La prioridad 0*, será dinámica y de tiempo compartido. Para esta prioridad se implementa una Round Robin.

Por cada pulsación de reloj la prioridad del proceso en ejecución se decrementa en 1, y cuando se le acabe el quantum se irá a la cola que le corresponda según su prioridad.

Si Algún proceso llega a la prioridad 0, se recalcula la prioridad de todos los procesos.

Recálculo, consiste en dividir entre dos la prioridad actual y sumarle la prioridad base.

De esta forma habrá:

- A ) Procesos que compiten por el procesador, que van de 0 a la prioridad base.
- B ) Procesos que usan mucho la E/S, a los cuales se les va incrementando la prioridad.

Cuando tenemos que pasar de un proceso de tiempo real a un proceso de tiempo compartido, no está especificado que se debe hacer, por lo que se deja a libre albedrío.

**9.7. WINDOWS NT Y WINDOWS 2.000**

Son Sistemas Operativos orientados a objetos y su unidad de planificación es el hilo.

a) La Prioridad

Define 32 niveles de prioridad (0-31), de las cuales:

- *Del 31 al 16*, serán estáticas y de tiempo real.
- *Del 15 al 1*, serán dinámicas.
- *Prioridad 0*, es el nivel del sistema.

Cada hilo que termine su quantum baja de prioridad.

Los hilos que se bloqueen, aumentarán su prioridad de alguna forma no especificada.

b) La política

Todos se ejecutan por Round Robin donde el quantum depende del nivel de prioridad.

Funcionamiento:

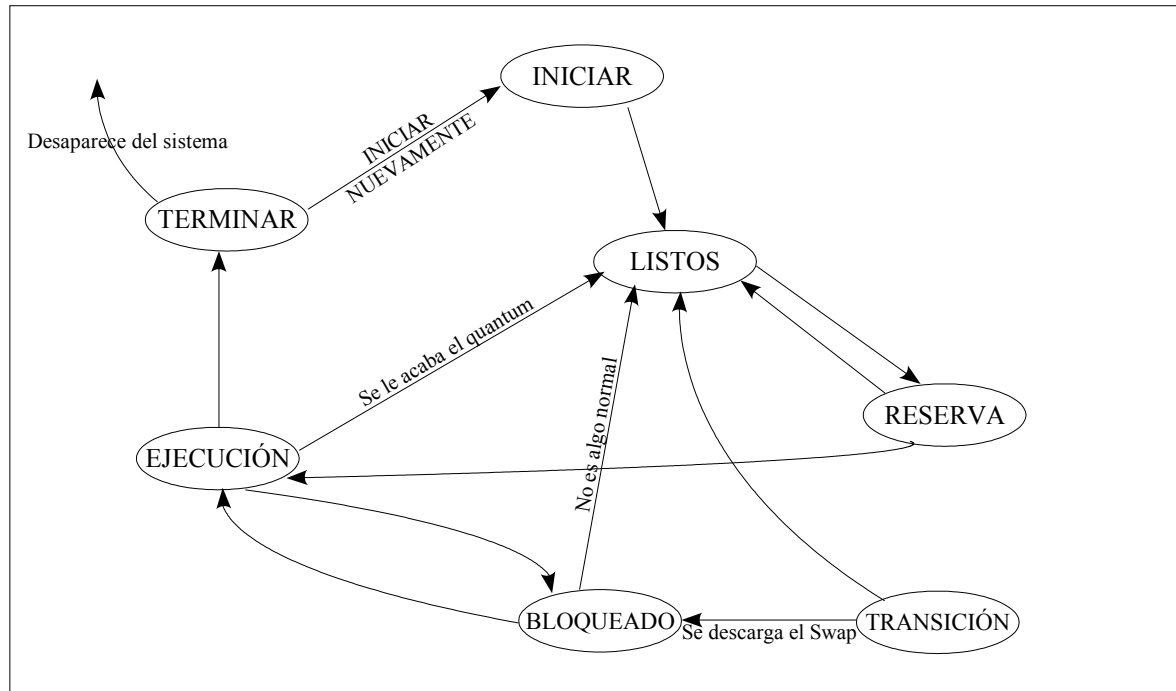
- Al hilo más prioritario, se le da el quantum de su prioridad (lo cual, dijimos, no está especificado)
- La prioridad entre colas es apropiativa.

c) Estados en los que pueden estar los hilos.

- Listo o preparado (como siempre).
- Reserva: donde está el próximo hilo que se vaya a despachar.
- Ejecución:

- En Transición (preparado suspendido): Preparado pero también descargado en la memoria Swap.
- Finalizado: Los procesos que han terminado pero pueden ser relanzados.

d) Diagrama de estado.



Esquema 17. Diagrama de estado en Windows NT y Windows 2.000

## 10.- MECANISMOS DE COMUNICACIÓN Y SINCRONIZACIÓN.

Los hilos y procesos tienen 3 formas de interactuar.

a) Sincronización.

Son los mecanismos necesarios para evitar el acceso concurrente de forma incorrecta a los recursos compartidos.

b) Señalización.

Son mecanismos orientados a la comunicación entre procesos, de modo que, un proceso avisa a otro (sin información) de un evento que está esperando.

c) Comunicación.

Son mecanismos para el envío de información de un proceso a otro.