**Procesos y Administración del Procesador**

1. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Introducción y Definiciones Sobre Procesos](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#IDSP)
2. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Estados de Procesos](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#EP)
3. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Procesamiento de Interrupciones](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PI)
4. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [El Núcleo del Sistema Operativo](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#NSO)
5. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Planificación de Procesos](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PP)
6. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Niveles de Planificación del Procesador](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#NPP)
7. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Objetivos de la Planificación](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#OP)
8. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Criterios de Planificación](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#CP)
9. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Planificación Apropiativa Versus No Apropiativa](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PAVNA)
10. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Temporizador de Intervalos o Reloj de Interrupción](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#TIRI)
11. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Prioridades](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#Prior)
12. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Tipos de Planificación](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#TP)
    1. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Planificación a Plazo Fijo](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PPF)
    2. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Planificación Garantizada](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PG)
    3. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Planificación del Primero en Entrar Primero en Salir (FIFO)](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#FIFO)
    4. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Planificación de Asignación en Rueda (RR: Round Robin)](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#RR)
    5. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Tamaño del Cuanto o Quantum](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#TCQ)
    6. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Planificación del Trabajo Más Corto Primero (SJF)](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#SJF)
    7. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Planificación del Tiempo Restante Más Corto (SRT)](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#SRT)
    8. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Planificación el Siguiente con Relación de Respuesta Máxima (HRN)](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#HRN)
    9. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Planificación por Prioridad](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PlPr)
    10. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Colas de Retroalimentación de Niveles Múltiples](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#CRNM)
    11. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Política Versus Mecanismo de Planificación](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PVMP)
    12. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif [Planificación de Dos Niveles](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PDN)
13. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Multiprocesamiento](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#Multi)
    1. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Introducción](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#Introduc)
    2. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Confiabilidad](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#Confi)
    3. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Explotación del Paralelismo](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#EP)
    4. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Paralelismo Masivo](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#PaMa)
    5. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Metas de los Sistemas de Multiprocesamiento](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#MSM)
    6. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Detección Automática del Paralelismo](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#DAP)
    7. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Distribución de Ciclos](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#DdC)
    8. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Reducción de la Altura del Árbol](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#RAA)
14. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Organización del Hardware del Multiprocesador](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#OHM)
    1. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Tiempo Compartido o Bus Común (o Conductor Común)](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#TCBC)
    2. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Matriz de Barras Cruzadas e Interruptores](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#MBCI)
    3. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Almacenamiento de Interconexión Múltiple](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#AIM)
15. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Grados de Acoplamiento en Multiprocesamiento](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#GAM)
    1. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Organización Maestro / Satélite](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#OMS)
16. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Sistema Operativo de Multiprocesadores](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#SOMP)
    1. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Maestro / Satélite](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#MS)
    2. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Ejecutivos Separados](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#ES)
    3. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0009.gif   [Tratamiento Simétrico](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#TrSi)
17. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Rendimiento del Sistema de Multiprocesamiento](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#RSM)
18. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Recuperación de Errores](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#RecEr)
19. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Multiprocesamiento Simétrico (MPS)](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#MPS)
20. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Tendencias de los Multiprocesadores](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#TenMul)
21. http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Bot0008.gif   [Fin](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#Fin)

**Introducción y Definiciones Sobre Procesos**

El concepto central de cualquier Sistema Operativo es el de **proceso**: una abstracción de un programa en ejecución también llamada **tarea.**

No hay un acuerdo universal sobre una definición de proceso, pero sí algunas definiciones aceptadas [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7):

* Un programa que se está ejecutando.
* Una actividad asincrónica.
* El **emplazamiento del control** de un procedimiento que está siendo ejecutado.
* Aquello que se manifiesta por la existencia en el Sistema Operativo de un **bloque de control de proceso**.
* Aquella entidad a la cual son asignados los procesadores.
* La unidad **despachable.**

En sistemas de **multiprogramación** la cpu alterna de programa en programa, en un esquema de **seudo paralelismo** , es decir que la cpu ejecuta en cierto instante un solo programa, intercambiando muy rápidamente entre uno y otro.

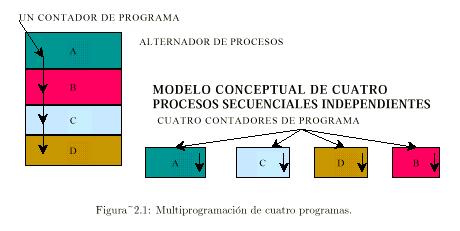
El **paralelismo real de hardware** se da en las siguientes situaciones:

* En ejecución de instrucciones de programa con más de un procesador de instrucciones en uso simultáneamente.
* Con la superposición de ejecución de instrucciones de programa con la ejecución de una o más operaciones de entrada / salida.

El objetivo es aumentar el paralelismo en la ejecución.

El **modelo de procesos** posee las siguientes características:

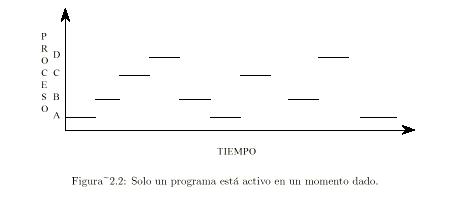
* Todo el software ejecutable, inclusive el Sistema Operativo, se organiza en varios **procesos secuenciales** o **procesos**.
* Un proceso incluye al programa en ejecución y a los valores activos del contador, registros y variables del mismo.
* Conceptualmente cada proceso tiene su propia cpu virtual.
* Si la cpu alterna entre los procesos, la velocidad a la que ejecuta un proceso no será uniforme, por lo que es necesario aclarar lo siguiente:
  + Que los procesos no deben programarse con hipótesis implícitas acerca del tiempo.
  + Que normalmente la mayoría de los procesos no son afectados por la multiprogramación subyacente de la cpu o las velocidades relativas de procesos distintos.
* Un proceso es una actividad de un cierto tipo, que tiene un programa, entrada, salida y estado.
* Un solo procesador puede ser compartido entre varios procesos con cierto *“algoritmo de planificación”* , el cual determina cuándo detener el trabajo en un proceso y dar servicio a otro distinto (ver Figura 2.1 [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23)).

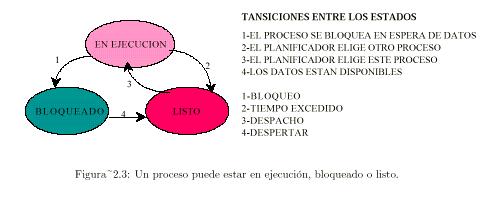


En cuanto a las **jerarquías de procesos** es necesario señalar que los Sistemas Operativos deben disponer de una forma de crear y destruir procesos cuando se requiera durante la operación, teniendo además presente que los procesos pueden generar procesos hijos mediante llamadas al Sistema Operativo, pudiendo darse ejecución en paralelo.

Respecto de los **estados del proceso** deben efectuarse las siguientes consideraciones:

* Cada proceso es una entidad independiente pero frecuentemente debe interactuar con otros procesos (ver Figura 2.2 [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23)).
* Los procesos pueden bloquearse en su ejecución porque:
  + Desde el punto de vista lógico no puede continuar porque espera datos que aún no están disponibles.
  + El Sistema Operativo asignó la cpu a otro proceso.
* Los estados [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23) que puede tener un proceso son (ver Figura 2.3 [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23)):
  + **En ejecución:** utiliza la cpu en el instante dado.
  + **Listo:** ejecutable, se detiene en forma temporal para que se ejecute otro proceso.
  + **Bloqueado:** *no se puede ejecutar* debido a la ocurrencia de algún evento externo.
* Son posibles cuatro **transiciones** entre estos estados.





**Estados de Procesos**

Durante su existencia un proceso pasa por una serie de estados discretos, siendo varias las circunstancias que pueden hacer que el mismo cambie de estado.

Debido a ello se puede establecer una “Lista de Listos” para los procesos “listos” y una “Lista de Bloqueados” para los “bloqueados”.

La *“Lista de Listos”* se mantiene en orden prioritario y la *“Lista de Bloqueados”* está desordenada, ya que los procesos se desbloquean en el orden en que tienen lugar los eventos que están esperando.

Al admitirse un trabajo en el sistema se crea un proceso equivalente y es insertado en la última parte de la *“Lista de Listos”.*

La asignación de la cpu al primer proceso de la *“Lista de Listos”* se denomina “Despacho”, que es ejecutado por una entidad del Sistema Operativo llamada *“Despachador”.*

El “Bloqueo” es la única transición de estado iniciada por el propio proceso del usuario, puesto que las otras transiciones son iniciadas por entidades ajenas al proceso.

La manifestación de un proceso en un Sistema Operativo es un *“Bloque de Control de Proceso”* (PCB) con información que incluye [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7):

* Estado actual del proceso.
* Identificación única del proceso.
* Prioridad del proceso.
* Apuntadores para localizar la memoria del proceso.
* Apuntadores para asignar recursos.
* Área para preservar registros.

Cuando el Sistema Operativo cambia la atención de la cpu entre los procesos, utiliza las áreas de preservación del PCB para mantener la información que necesita para reiniciar el proceso cuando consiga de nuevo la cpu.

Los sistemas que administran los procesos deben poder crear, destruir, suspender, reanudar, cambiar la prioridad, bloquear, despertar y despachar un proceso.

La *“creación”* de un proceso significa:

* Dar nombre al proceso.
* Insertar un proceso en la lista del sistema de procesos conocidos.
* Determinar la prioridad inicial del proceso.
* Crear el bloque de control del proceso.
* Asignar los recursos iniciales del proceso.

Un proceso puede *“crear un nuevo proceso”*, en cuyo caso el proceso creador se denomina *“proceso padre”* y el proceso creado *“proceso hijo”* y se obtiene una *“estructura jerárquica de procesos”.*

La *“destrucción”* de un proceso implica:

* Borrarlo del sistema.
* Devolver sus recursos al sistema.
* Purgarlo de todas las listas o tablas del sistema.
* Borrar su bloque de control de procesos.

Un proceso *“suspendido”* no puede proseguir hasta que otro proceso lo reanude.

Reanudar (reactivar) un proceso implica reiniciarlo en el punto donde fue suspendido.

La *“destrucción”* de un proceso puede o no significar la destrucción de los procesos hijos, según el Sistema Operativo.

Generalmente se denomina *“Tabla de Procesos”* al conjunto de información de control sobre los distintos procesos.

**Procesamiento de Interrupciones**

Una “interrupción” es un evento que altera la secuencia en que el procesador ejecuta las instrucciones; es un hecho generado por el hardware del computador [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7).

Cuando ocurre una interrupción, el Sistema Operativo:

* Obtiene el control.
* Salva el estado del proceso interrumpido, generalmente en su bloque de control de procesos.
* Analiza la interrupción.
* Transfiere el control a la rutina apropiada para la manipulación de la interrupción.

Una interrupción puede ser iniciada por un proceso en estado de ejecución o por un evento que puede o no estar relacionado con un proceso en ejecución.

Generalmente las **interrupciones** se pueden clasificar por **tipos** según el siguiente detalle (ver Tabla 2.1 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7):

* *“SVC (llamada al supervisor)”:* es una petición generada por el usuario para un servicio particular del sistema, por ejemplo, realización de Entrada / Salida u obtención de más memoria.
* *“Entrada / Salida”:* son iniciadas por el hardware de Entrada / Salida, indicando a la cpu que ha cambiado el estado de un canal o dispositivo, por ejemplo, finalización de Entrada / Salida u ocurrencia de un error.
* *“Externas”:* son causadas por distintos eventos, por ejemplo, expiración de un cuanto en un reloj de interrupción o recepción de una señal de otro procesador en un sistema multiprocesador.
* *“De reinicio”:* ocurren al presionar la “tecla de reinicio” o cuando llega una instrucción de reinicio de otro procesador en un sistema multiprocesador.
* *“De verificación de programa”:* son causadas por errores producidos durante la ejecución de procesos, por ejemplo:
  + Un intento de dividir por cero.
  + Un intento de un proceso de usuario de ejecutar una instrucción privilegiada.
  + Un intento de ejecutar un código de operación inválido.
* *“De verificación de máquina”:* son ocasionadas por un mal funcionamiento del hardware.

|  |  |
| --- | --- |
| **Tipo de Interrupción** | **Descripción** |
| SVC | Llamada al Sistema Operativo |
| Entrada / Salida | Cambio de estado de un canal o dispositivo |
| Externa | Evento externo al sistema |
| De Reinicio | Reinicio del procesamiento |
| De Verificación de Programa | Errores de procesos |
| De Verificación de Máquina | Errores de hardware |
| **Tabla 2.1: Tipos de interrupciones.** | |

El Sistema Operativo incluye rutinas llamadas **“Manipuladores de Interrupciones (IH)”** para procesar cada tipo diferente de interrupción.   
Cuando se produce una interrupción el Sistema Operativo efectúa las siguientes acciones:

* Salva el estado del proceso interrumpido.
* Dirige el control al manipulador de interrupciones adecuado.
* Se aplica la técnica de “Cambio de Contexto” .

Los Sistemas Operativos instrumentan información de control que puede aparecer como las “Palabras de Estado de Programa (PSW)”, las cuales controlan el orden de ejecución de las instrucciones y contienen información sobre el estado del proceso.

Existen tres tipos de PSW, que son la “actual”, la “nueva” y la “vieja”.

La “PSW Actual” almacena la dirección de la próxima instrucción que será ejecutada e indica los tipos de instrucciones actualmente “habilitadas” e inhabilitadas”.

En un **sistema uniprocesador** existe:

* Solo una PSW actual.
* Seis PSW nuevas (una para cada tipo de interrupción).
* Seis PSW viejas (una para cada tipo de interrupción).

La PSW nueva para un tipo de interrupción dado contiene la dirección en el hardware donde reside el manipulador de interrupciones para este tipo específico.

Cuando ocurre una interrupción para la cual el procesador no está inhabilitado, ocurren las siguientes acciones:

* El hardware cambia las PSW en los casos siguientes:
  + Al almacenar la PSW actual en la PSW vieja, para este tipo de interrupción.
  + Al almacenar la PSW nueva en la PSW actual, para este tipo de interrupción.
* Luego de este “intercambio de PSW”:
  + La PSW actual contiene la dirección del manipulador de interrupción adecuado.
  + El manipulador de interrupciones procesa la interrupción.
  + Luego de procesar la interrupción, la cpu es enviada al:
    - Proceso que estaba en ejecución en el momento de la interrupción, o al
    - Proceso de listo de más alta prioridad.
  + La acción precedente depende de si el proceso de interrupción es:
    - **“Apropiativo”:** obtiene la cpu solo si no hay procesos de listos.
    - **“No apropiativo”:** obtiene de nuevo la cpu.

**El Núcleo del Sistema Operativo**

El *“núcleo”* del Sistema Operativo controla todas las operaciones que implican procesos y representa solo una pequeña porción del código de todo el Sistema Operativo pero es de amplio uso [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7).

Generalmente permanece en el almacenamiento primario.

El proceso de interrupciones se incluye en el núcleo ya que debe ser rápido (especialmente en sistemas multiusuario), para optimizar el uso de los recursos del sistema y proveer tiempos de respuesta aceptables a los usuarios interactivos.

El **núcleo** inhabilita las interrupciones mientras responde a una interrupción. Las interrupciones son habilitadas de nuevo después de completar el proceso de una interrupción.

El **núcleo** del Sistema Operativo generalmente realiza las siguientes **funciones:**

* Manipulación de interrupciones.
* Creación y destrucción de procesos.
* Cambio de estados de procesos.
* Despacho.
* Suspensión y reanudación de procesos.
* Sincronización de procesos.
* Comunicación entre procesos.
* Manipulación de bloques de control de proceso.
* Soporte de las actividades de Entrada / Salida.
* Soporte de la asignación y desasignación de almacenamiento.
* Soporte del sistema de archivos.
* Soporte de un mecanismo de llamada / regreso al procedimiento.
* Soporte de ciertas funciones contables (estadísticas) del sistema.

**Planificación de Procesos**

Cuando más de un proceso es ejecutable desde el punto de vista lógico, el Sistema Operativo debe decidir cuál de ellos debe ejecutarse en primer término.

El **Planificador** es la porción del Sistema Operativo que decide y el **Algoritmo de Planificación** es el utilizado.

Los principales *“criterios”* respecto de un buen algoritmo de planificación [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23) son la equidad, la eficacia, el tiempo de respuesta, el tiempo de regreso y el rendimiento (ver Tabla 2.2 [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23)).

|  |  |
| --- | --- |
| **Criterio** | **Descripción** |
| Equidad | Garantizar que cada proceso obtiene su proporción justa de la cpu |
| Eficacia | Mantener ocupada la cpu el ciento por ciento del tiempo |
| Tiempo de respuesta | Minimizar el tiempo de respuesta para los usuarios interactivos |
| Tiempo de regreso | Minimizar el tiempo que deben esperar los usuarios por lotes (batch) para obtener sus resultados |
| Rendimiento | Maximizar el número de tareas procesadas por hora |
| **Tabla 2.2: Criterios de un buen algoritmo de planificación.** | |

Algunas de estas metas son contradictorias, por ejemplo, minimizar el tiempo de respuesta para los usuarios interactivos significaría no ejecutar las tareas batch.

Cada proceso es único e impredecible, es decir que pueden requerir intensivamente operaciones de Entrada / Salida o intensivamente cpu; el planificador del Sistema Operativo no tiene la certeza de cuánto tiempo transcurrirá hasta que un proceso se bloquee, ya sea por una operación de Entrada / Salida o por otra razón .

Para evitar que un proceso se apropie de la cpu un tiempo excesivo, los equipos poseen un dispositivo que provoca una interrupción en forma periódica, por ejemplo 60 hz, o sea sesenta veces por segundo.

En cada interrupción del reloj el Sistema Operativo decide si el proceso que se está ejecutando continúa o si el proceso agotó su tiempo de cpu y debe suspenderse y ceder la cpu a otro proceso.

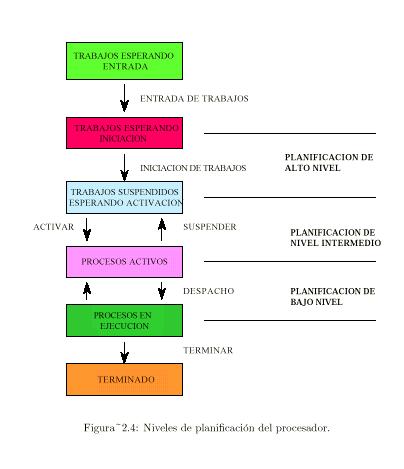
Los principales conceptos relacionados con **Planificación del Procesador** son los siguiente:

* *Planificación apropiativa:* es la estrategia de permitir que procesos ejecutables (desde el punto de vista lógico) sean suspendidos temporalmente.
* *Planificación no apropiativa:* es la estrategia de permitir la ejecución de un proceso hasta terminar.
* *Planificación del procesador:* determinar cuándo deben asignarse los procesadores y a qué procesos, lo cual es responsabilidad del Sistema Operativo.

**Niveles de Planificación del Procesador**

Se consideran tres niveles importantes de planificación, los que se detallan a continuación (ver Figura 2.4 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)):

* *Planificación de alto nivel:*
  + También se denomina Planificación de trabajos.
  + Determina a qué trabajos se les va a permitir competir activamente por los recursos del sistema, lo cual se denomina Planificación de admisión.
* *Planificación de nivel intermedio:*
  + Determina a qué procesos se les puede permitir competir por la cpu.
  + Responde a fluctuaciones a corto plazo en la carga del sistema y efectúa “suspensiones” y “activaciones” (“reanudaciones”) de procesos.
  + Debe ayudar a alcanzar ciertas metas en el rendimiento total del sistema.
* *Planificación de bajo nivel:*
  + Determina a qué proceso listo se le asigna la cpu cuando esta queda disponible y asigna la cpu al mismo, es decir que “despacha” la cpu al proceso.
  + La efectúa el **Despachador** del Sistema Operativo, el que opera muchas veces por segundo y reside siempre en el almacenamiento primario.



Los distintos Sistemas Operativos utilizan varias **Políticas de Planificación**, que se instrumentan mediante **Mecanismos de Planificación**.

**Objetivos de la Planificación**

Los objetivos de la planificación del procesador son los siguientes e involucran a los conceptos detallados seguidamente [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7):

* **Ser justa:**
  + Todos los procesos son tratados de igual manera.
  + Ningún proceso es postergado indefinidamente.
* **Maximizar la capacidad de ejecución:**
  + Maximizar el número de procesos servidos por unidad de tiempo.
* **Maximizar el número de usuarios interactivos que reciban unos tiempos de respuesta aceptables:**
  + En un máximo de unos segundos.
* **Ser predecible:**
  + Un trabajo dado debe ejecutarse aproximadamente en la misma cantidad de tiempo independientemente de la carga del sistema.
* **Minimizar la sobrecarga:**
  + No suele considerarse un objetivo muy importante.
* **Equilibrar el uso de recursos:**
  + Favorecer a los procesos que utilizarán recursos infrautilizados.
* **Equilibrar respuesta y utilización:**
  + La mejor manera de garantizar buenos tiempos de respuesta es disponer de los recursos suficientes cuando se necesitan, pero la utilización total de recursos podrá ser pobre.
* **Evitar la postergación indefinida:**
  + Se utiliza la estrategia del “envejecimiento” .
  + Mientras un proceso espera por un recurso su prioridad debe aumentar, así la prioridad llegará a ser tan alta que el proceso recibirá el recurso esperado.
* **Asegurar la prioridad:**
  + Los mecanismos de planificación deben favorecer a los procesos con prioridades más altas.
* **Dar preferencia a los procesos que mantienen recursos claves:**
  + Un proceso de baja prioridad podría mantener un recurso clave, que puede ser requerido por un proceso de más alta prioridad.
  + Si el recurso es no apropiativo, el mecanismo de planificación debe otorgar al proceso un tratamiento mejor del que le correspondería normalmente, puesto que es necesario liberar rápidamente el recurso clave.
* **Dar mejor tratamiento a los procesos que muestren un “comportamiento deseable”:**
  + Un ejemplo de comportamiento deseable es una tasa baja de paginación.
* **Degradarse suavemente con cargas pesadas:**
  + Un mecanismo de planificación no debe colapsar con el peso de una exigente carga del sistema.
  + Se debe evitar una carga excesiva mediante las siguientes acciones:
    - No permitiendo que se creen nuevos procesos cuando la carga ya es pesada.
    - Dando servicio a la carga más pesada al proporcionar un nivel moderadamente reducido de servicio a todos los procesos.

Muchas de estas metas se encuentran en conflicto entre sí, por lo que la planificación se convierte en un problema complejo.

**Criterios de Planificación**

Para realizar los objetivos de la planificación, un mecanismo de planificación debe considerar lo siguiente [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7):

* La limitación de un proceso a las operaciones de Entrada / Salida: cuando un proceso consigue la cpu, ¿la utiliza solo brevemente antes de generar una petición de Entrada / Salida?.
* La limitación de un proceso a la cpu: cuando un proceso obtiene la cpu, ¿tiende a usarla hasta que expira su tiempo?.
* Si un proceso es por lote (batch) o interactivo: los usuarios interactivos deben recibir inmediato servicio para garantizar buenos tiempos de respuesta.
* ¿Qué urgencia tiene una respuesta rápida?: por ejemplo, un proceso de tiempo real de un sistema de control que supervise una refinería de combustible requiere una respuesta rápida, más rápida que la respuesta requerida por un proceso en lotes (batch) que deberá entregarse al día siguiente.
* La prioridad de un proceso: a mayor prioridad mejor tratamiento.
* Frecuentemente un proceso genera fallos (carencias) de página:
  + Probablemente los procesos que generan pocos fallos de página hayan acumulado sus “conjuntos de trabajo” en el almacenamiento principal.
  + Los procesos que experimentan gran cantidad de fallos de página aún no han establecido sus conjuntos de trabajo.
  + Un criterio indica favorecer a los procesos que han establecido sus conjuntos de trabajo.
  + Otro criterio indica favorecer a los procesos con una tasa alta de fallos de página ya que rápidamente generarán una petición de Entrada / Salida.
* Frecuentemente un proceso ha sido apropiado por otro de más alta prioridad, lo cual significa lo siguiente:
  + A menudo los procesos apropiados deben recibir un tratamiento menos favorable.
  + Cada vez que el Sistema Operativo asume la sobrecarga para hacer ejecutar este proceso, el corto tiempo de ejecución antes de la apropiación no justifica la sobrecarga de hacer ejecutar al proceso en primer lugar.
* ¿Cuánto tiempo de ejecución real ha recibido el proceso?: un criterio considera que debe ser favorecido un proceso que ha recibido muy poco tiempo de cpu.
* ¿Cuánto tiempo adicional va a necesitar el proceso para terminar?: los tiempos promedio de espera pueden reducirse priorizando los procesos que requieren de un tiempo de ejecución mínimo para su terminación, pero pocas veces es posible conocer la cantidad de tiempo adicional que cada proceso necesita para terminar.

**Planificación Apropiativa Versus No Apropiativa**

Las Disciplinas de Planificación pueden ser **Apropiativas** o **No Apropiativas** (ver Tabla 2.3 [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23)).

|  |  |
| --- | --- |
| **Disciplina** | **Descripción** |
| “Apropiativa” | Una vez que se le ha otorgado la cpu a un proceso, le puede ser retirada |
| “No Apropiativa” | Una vez que se le ha otorgado la cpu a un proceso, no le puede ser retirada |
| **Tabla 2.3:  Disciplinas de planificación del procesador.** | |

Las principales características de la **planificación apropiativa** son las siguientes:

* Es útil cuando los procesos de alta prioridad requieren atención rápida.
* Es importante para garantizar buenos tiempos de respuesta en sistemas interactivos de tiempo compartido.
* Tiene su costo en recursos, ya que el intercambio de contexto implica sobrecarga y además requiere mantener muchos procesos en el almacenamiento principal, en espera de la cpu, lo que también implica sobrecarga.

Las principales características de la **planificación no apropiativa** son las siguientes:

* Significa que los trabajos “largos” hacen esperar a los trabajos “cortos”.
* Logra más equidad en el tratamiento de los procesos.
* Logra hacer más predecibles los tiempos de respuesta puesto que los trabajos nuevos de prioridad alta no pueden desplazar a los trabajos en espera.

El diseño de un mecanismo apropiativo hace necesario considerar las arbitrariedades de casi cualquier esquema de prioridades, en razón de que muchas veces las propias prioridades no son asignadas de forma significativa [[25, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#25).

El mecanismo debería ser sencillo pero efectivo y significativo.

**Temporizador de Intervalos o Reloj de Interrupción**

El proceso al cual está asignada la cpu se dice que está en ejecución y puede ser un proceso de Sistema Operativo o de usuario.

El Sistema Operativo dispone de mecanismos para quitarle la cpu a un proceso de usuario para evitar que monopolice el sistema.

El Sistema Operativo posee un *“reloj de interrupción”* o *“temporizador de intervalos”* para generar una interrupción, en algún tiempo futuro específico o después de un transcurso de tiempo en el futuro; la cpu es entonces despachada hacia el siguiente proceso [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7).

Un proceso retiene el control de la cpu hasta que ocurra alguna de las siguientes situaciones:

* La libera voluntariamente.
* El reloj la interrumpe.
* Alguna otra interrupción atrae la atención de la cpu.

Si el reloj interrumpe un proceso de usuario, la interrupción causa la ejecución del Sistema Operativo, el que decide cuál será el proceso que obtendrá la cpu.

El reloj de interrupción ayuda a garantizar tiempos de respuesta razonables a usuarios interactivos, ya que evita que el sistema se “cuelgue” a un solo usuario en un ciclo infinito y permite que los procesos respondan a “eventos dependientes del tiempo”.

Asimismo, los procesos que necesitan una ejecución periódica dependen del reloj de interrupción [[22, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#22).

**Prioridades**

Las **prioridades** pueden ser de distinto tipo (ver Tabla 2.4 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)).

|  |
| --- |
| **Tipos de prioridades** |
| Asignadas automáticamente por el sistema |
| Asignadas desde el exterior |
| Dinámicas |
| Estáticas |
| Asignadas racionalmente |
| Asignadas arbitrariamente |
| **Tabla 2.4: Tipos de prioridades.** |

En el caso de prioridades asignadas arbitrariamente, un mecanismo del sistema necesita distinguir entre procesos sin importarle cuál es el más importante.

Las principales características de las **prioridades estáticas** son las siguientes:

* No cambian.
* Los mecanismos de implementación son sencillos.
* Implican una sobrecarga relativamente baja.
* No responden a cambios en el ambiente (contexto) que harían deseable ajustar alguna prioridad.

Las principales características de las **prioridades dinámicas** son las siguientes:

* Responden al cambio.
* La prioridad inicial asignada a un proceso puede durar poco tiempo, luego se la reajusta a un mejor valor.
* Los mecanismos de implementación son más complicados que para prioridades estáticas.
* Implican una sobrecarga mayor que para esquemas estáticos.

Respecto de las **prioridades adquiridas**, se hace referencia al tratamiento especial que en situaciones excepcionales requiere un cierto proceso, lo que puede significar restar recursos a los demás procesos.

**Tipos de Planificación**

**Planificación a Plazo Fijo**

Ciertos trabajos se planifican para ser terminados en un tiempo específico o plazo fijo. Es una planificación compleja debido a los siguientes factores:

* El usuario debe suministrar anticipadamente una lista precisa de recursos necesarios para el proceso, pero generalmente no se dispone de dicha información.
* La ejecución del trabajo de plazo fijo no debe producir una grave degradación del servicio a otros usuarios.
* El sistema debe planificar cuidadosamente sus necesidades de recursos hasta el plazo fijo, lo que se puede complicar con las demandas de recursos de nuevos procesos que ingresen al sistema.
* La concurrencia de varios procesos de plazo fijo (activos a la vez) puede requerir métodos sofisticados de optimización.
* La administración intensiva de recursos puede generar una considerable sobrecarga adicional.

**Planificación Garantizada**

Se establecen compromisos de desempeño con el proceso del usuario, por ejemplo, si existen *“n”* procesos en el sistema, el proceso del usuario recibirá cerca del *“1 / n”* de la potencia de la cpu.

El sistema debe tener un registro del tiempo de cpu que cada proceso ha tenido desde su entrada al sistema y del tiempo transcurrido desde esa entrada.

Con los datos anteriores y el registro de procesos en curso de ejecución, el sistema calcula y determina qué procesos están más alejados por defecto de la relación *“1 / n”* prometida y prioriza los procesos que han recibido menos cpu de la prometida.

**Planificación del Primero en Entrar Primero en Salir (FIFO)**

Es muy simple, los procesos se despachan de acuerdo con su tiempo de llegada a la cola de listos.

Una vez que el proceso obtiene la cpu, se ejecuta hasta terminar, ya que es una disciplina “no apropiativa”.

Puede ocasionar que procesos largos hagan esperar a procesos cortos y que procesos no importantes hagan esperar a procesos importantes.

Es más predecible que otros esquemas.

No puede garantizar buenos tiempos de respuesta interactivos.

Suele utilizarse integrado a otros esquemas, por ejemplo, de la siguiente manera:

* Los procesos se despachan con algún esquema de prioridad.
* Los procesos con igual prioridad se despachan “FIFO”.

**Planificación de Asignación en Rueda (RR: Round Robin)**

Los procesos se despachan en “FIFO” y disponen de una cantidad limitada de tiempo de cpu, llamada “división de tiempo” o “cuanto”.

Si un proceso no termina antes de expirar su tiempo de cpu ocurren las siguientes acciones:

1. La cpu es apropiada.
2. La cpu es otorgada al siguiente proceso en espera.
3. El proceso apropiado es situado al final de la lista de listos.

Es efectiva en ambientes de tiempo compartido.

La sobrecarga de la apropiación se mantiene baja mediante mecanismos eficientes de intercambio de contexto y con suficiente memoria principal para los procesos.

**Tamaño del Cuanto o Quantum**

La determinación del tamaño del cuanto es decisiva para la operación efectiva de un sistema computacional [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7).

Los interrogantes son: ¿cuanto pequeño o grande?, ¿cuanto fijo o variable? y ¿cuanto igual para todos los procesos de usuarios o determinado por separado para cada uno de ellos?.

Si el cuanto se hace muy grande, cada proceso recibe todo el tiempo necesario para llegar a su terminación, por lo cual la asignación en rueda (“RR”) degenera en “FIFO”.

Si el cuanto se hace muy pequeño, la sobrecarga del intercambio de contexto se convierte en un factor dominante y el rendimiento del sistema se degrada, puesto que la mayor parte del tiempo de cpu se invierte en el intercambio del procesador (cambio de contexto) y los procesos de usuario disponen de muy poco tiempo de cpu.

El cuanto debe ser lo suficientemente grande como para permitir que la gran mayoría de las peticiones interactivas requieran de menos tiempo que la duración del cuanto, es decir que el tiempo transcurrido desde el otorgamiento de la cpu a un proceso hasta que genera una petición de Entrada / Salida debe ser menor que el cuanto establecido, de esta forma, ocurrida la petición la cpu pasa a otro proceso y como el cuanto es mayor que el tiempo transcurrido hasta la petición de Entrada / Salida, los procesos trabajan al máximo de velocidad, se minimiza la sobrecarga de apropiación y se maximiza la utilización de la   
Entrada / Salida.

El cuanto óptimo varía de un sistema a otro y con la carga, siendo un valor de referencia 100 mseg (cien milisegundos).

**Planificación del Trabajo Más Corto Primero (SJF)**

Es una disciplina no apropiativa y por lo tanto no recomendable en ambientes de tiempo compartido.

El proceso en espera con el menor tiempo estimado de ejecución hasta su terminación es el siguiente en ejecutarse.

Los tiempos promedio de espera son menores que con “FIFO”.

Los tiempos de espera son menos predecibles que en “FIFO”.

Favorece a los procesos cortos en detrimento de los largos.

Tiende a reducir el número de procesos en espera y el número de procesos que esperan detrás de procesos largos.

Requiere un conocimiento preciso del tiempo de ejecución de un proceso, lo que generalmente se desconoce.

Se pueden estimar los tiempos en base a series de valores anteriores.

**Planificación del Tiempo Restante Más Corto (SRT)**

Es la contraparte apropiativa del SJF.

Es útil en sistemas de tiempo compartido.

El proceso con el tiempo estimado de ejecución menor para …nalizar es el siguiente en ser ejecutado.

Un proceso en ejecución puede ser apropiado por un nuevo proceso con un tiempo estimado de ejecución menor.

Tiene mayor sobrecarga que la planificación SJF.

Debe mantener un registro del tiempo de servicio transcurrido del proceso en ejecución, lo que aumenta la sobrecarga.

Los trabajos largos tienen un promedio y una varianza de los tiempos de espera aún mayor que en SJF.

La apropiación de un proceso a punto de terminar por otro de menor duración recién llegado podría significar un mayor tiempo de cambio de contexto (administración del procesador) que el tiempo de finalización del primero.

*Al diseñarse los Sistemas Operativos se debe considerar cuidadosamente la sobrecarga de los mecanismos de administración de recursos comparándola con los beneficios esperados.*

**Planificación el Siguiente con Relación de Respuesta Máxima (HRN)**

Corrige algunas de las debilidades del SJF, tales como el exceso de perjuicio hacia los procesos (trabajos) largos y el exceso de favoritismo hacia los nuevos trabajos cortos.

Es una disciplina no apropiativa.

La prioridad de cada proceso está en función no sólo del tiempo de servicio del trabajo, sino que también influye la cantidad de tiempo que el trabajo ha estado esperando ser servido.

Cuando un proceso ha obtenido la cpu, corre hasta terminar.

Las prioridades, que son dinámicas, se calculan según la siguiente fórmula, donde *pr* es la *“prioridad”*, *te* es el *“tiempo de espera”* y *ts* es el *“tiempo de servicio”:*



**Planificación por Prioridad**

Considera factores externos al proceso [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23).

Las ideas centrales son que cada proceso tiene asociada una prioridad y que el proceso ejecutable con máxima prioridad es el que tiene el permiso de ejecución.

Los procesos de alta prioridad podrían ejecutar indefinidamente, ya que el planificador del sistema puede disminuir la prioridad del proceso en ejecución en cada interrupción del reloj.

Las prioridades también pueden ser asignadas dinámicamente por el sistema para lograr ciertas metas relacionadas con el procesador o la Entrada / Salida.

Los procesos limitados por la Entrada / Salida (requerimientos intensivos de Entrada / Salida) ocupan mucho de su tiempo en espera de operaciones de Entrada / Salida, por lo tanto:

* Deben tener prioridad para usar la cpu y efectuar la siguiente petición de Entrada / Salida, ya que se ejecutará (la operación de Entrada / Salida) en paralelo con otro proceso que utilice la cpu.
* Si deben esperar mucho tiempo a la cpu estarán ocupando memoria por un tiempo innecesario.

Un algoritmo sencillo consiste en establecer que la prioridad sea *“1 / f”,* donde *“f”* es la fracción del último cuanto utilizado por el proceso.

Un proceso que utilice 2 mseg (dos milisegundos) de su cuanto de 100 mseg (cien milisegundos) tendrá prioridad 50 (cincuenta).

Un proceso que se ejecutó 50 mseg antes del bloqueo tendrá prioridad 2.

Un proceso que utilizó todo el cuanto tendrá prioridad 1.

Frecuentemente los procesos se agrupan en *“Clases de Prioridad”,* en cuyo caso se utiliza la Planificación con Prioridades entre las clases y con Round Robin (RR) dentro de cada clase. Si las prioridades no se reajustan en algún momento, los procesos de las clases de prioridad mínima podrían demorarse indefinidamente.

**Colas de Retroalimentación de Niveles Múltiples**

Proporcionan una estructura para lograr los siguientes objetivos:

* Favorecer trabajos cortos.
* Favorecer trabajos limitados por la Entrada / Salida para optimizar el uso de los dispositivos de Entrada / Salida.
* Determinar la naturaleza de un trabajo lo más rápido posible y planificar el trabajo (proceso) en consecuencia.

Un nuevo proceso entra en la red de línea de espera al final de la cola superior.

Se mueve por esta cola “FIFO” hasta obtener la cpu.

Si el trabajo termina o abandona la cpu para esperar por la terminación de una operación de Entrada / Salida o la terminación de algún otro suceso, el trabajo abandona la red de línea de espera.

Si su cuanto expira antes de abandonar la cpu voluntariamente, el proceso se coloca en la parte trasera de la cola del siguiente nivel inferior.

El trabajo recibe servicio al llegar a la cabeza de esta cola si la primera está vacía.

Mientras el proceso continúe consumiendo totalmente su cuanto en cada nivel, continuará moviéndose hacia el final de las colas inferiores.

Generalmente hay una cola en la parte más profunda a través de la cual el proceso circula en asignación de rueda hasta que termina.

Existen esquemas en los que el cuanto otorgado al proceso aumenta a medida que el proceso se mueve hacia las colas de los niveles inferiores, en tal caso, cuanto más tiempo haya estado el proceso en la red de línea de espera, mayor será su cuanto cada vez que obtiene la cpu y no podrá obtener la cpu muy a menudo debido a la mayor prioridad de los procesos de las colas superiores.

Un proceso situado en una cola dada no podrá ser ejecutado hasta que las colas de los niveles superiores estén vacías.

Un proceso en ejecución es apropiado por un proceso que llegue a una cola superior.

Es un mecanismo adaptable, es decir que se adapta a cargas variables.

A los efectos de una revisión gráfica de lo enunciado precedentemente, ver la figura 2.5 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7).

**Política Versus Mecanismo de Planificación**

Puede ocurrir que haya procesos con muchos procesos hijos ejecutándose bajo su control, por ejemplo, un proceso en un DBMS con procesos hijos atendiendo funciones específicas, tales como, análisis de interrogantes, acceso a discos, etc.

Es posible que el proceso principal (padre) pueda identificar la importancia (o criticidad) de sus procesos hijos, pero los planificadores analizados no aceptan datos de los procesos de usuario relativos a decisiones de planificación.

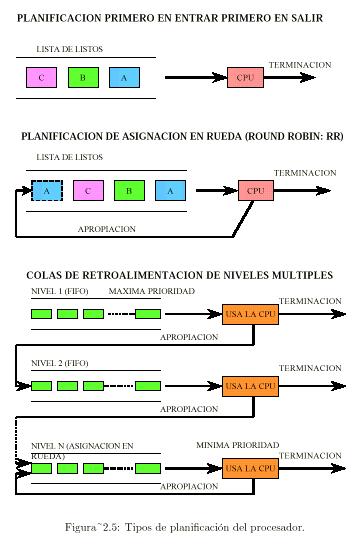
La solución es separar el **mecanismo de planificación** de la **política de planificación**, para ello se parametriza el algoritmo de planificación y los parámetros pueden ser determinados por medio de procesos del usuario; así el mecanismo está en el núcleo del Sistema Operativo pero la política queda establecida por un proceso del usuario.

**Planificación de Dos Niveles**

Los esquemas analizados hasta ahora suponen que todos los procesos ejecutables están en la memoria principal.

Si la memoria principal es insuficiente, ocurrirá lo siguiente [[23, Tanenbaum]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#23):

* Habrá procesos ejecutables que se mantengan en disco.
* Habrá importantes implicaciones para la planificación, tales como las siguientes:
  + El tiempo de alternancia entre procesos para traer y procesar un proceso del disco es considerablemente mayor que el tiempo para un proceso que ya está en la memoria principal.
  + Es más eficiente el intercambio de los procesos con un planificador de dos niveles.



El esquema operativo de un planificador de dos niveles es como sigue:

1. Se carga en la memoria principal cierto subconjunto de los procesos ejecutables.
2. El planificador se restringe a ellos durante cierto tiempo.
3. Periódicamente se llama a un planificador de nivel superior para efectuar las siguientes tareas:
   1. Eliminar de la memoria los procesos que hayan permanecido en ella el tiempo suficiente.
   2. Cargar a memoria los procesos que hayan estado en disco demasiado tiempo.
4. El planificador de nivel inferior se restringe de nuevo a los procesos ejecutables que se encuentren en la memoria.
5. El planificador de nivel superior se encarga de desplazar los procesos de memoria a disco y viceversa.

Los criterios que podría utilizar el planificador de nivel superior para tomar sus decisiones son los que se indican a continuación:

* ¿Cuánto tiempo ha transcurrido desde el último intercambio del proceso?.
* ¿Cuánto tiempo de cpu ha utilizado recientemente el proceso?.
* ¿Qué tan grande es el proceso? (generalmente los procesos pequeños no causan tantos problemas en este sentido).
* ¿Qué tan alta es la prioridad del proceso?.

El planificador de nivel superior podría utilizar cualquiera de los métodos de planificación analizados.

**Multiprocesamiento**

**Introducción**

Es una tendencia significativa en el campo de la computación.

Consiste en configurar un sistema de computación con varios procesadores .

No es un enfoque nuevo pero sí posee grandes perspectivas en función del desarrollo de los microprocesadores.

Se podrían concebir sistemas construidos por cientos o miles de microprocesadores.

**Inicio:[http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Ani0009.gif](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#Inicio)    Fin:**[http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Ani0006.gif](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SO2.htm#Fin)

**Confiabilidad**

Si un procesador falla, los restantes continúan operando, lo cual no es automático y requiere de un diseño cuidadoso.

Un procesador que falla habrá de informarlo a los demás de alguna manera, para que se hagan cargo de su trabajo.

Los procesadores en funcionamiento deben poder detectar el fallo de un procesador determinado.

El Sistema Operativo debe percibir que ha fallado un procesador determinado y ya no podrá asignarlo y también debe ajustar sus estrategias de asignación de recursos para evitar la sobrecarga del sistema que está degradado.

**Explotación del Paralelismo**

*La mayoría de los sistemas de multiprocesamiento tienen como meta principal el incremento de la capacidad de ejecución.*

La programación sigue siendo esencialmente secuencial y generalmente no se explota la concurrencia.

Las principales razones son las siguientes:

* Las personas piensan en forma secuencial.
* Ningún lenguaje humano proporciona la expresión adecuada de paralelismo, pero existen lenguajes de computación con soporte de concurrencia (por ejemplo, Ada, Pascal Concurrente, etc.).
* Ni el multiprocesamiento ha sido usado con amplitud para explotar el paralelismo.
* El hardware tradicional del computador está orientado hacia la operación secuencial.
* Es muy difícil depurar programas en paralelo.

Los multiprocesadores no se utilizan a menudo para explotar el paralelismo ya que es muy escaso el software que explote el paralelismo.

Lo deseable es que los Sistemas Operativos y compiladores puedan detectar e implementar el paralelismo automáticamente.

**Paralelismo Masivo**

Se debe disponer de suficientes procesadores como para que todas las operaciones que puedan ser ejecutadas en paralelo puedan ser asignadas a procesadores separados [[14, Pino y Marrone]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#14).

Esto ofrece una forma de ejecutar un programa en el menor tiempo posible.

La cuestión central es, disponiendo del paralelismo masivo, ¿cuál es el tiempo mínimo requerido para ejecutar un algoritmo determinado?.

**Metas de los Sistemas de Multiprocesamiento**

Las metas de los sistemas de multiprocesamiento generalmente son la confiabilidad y la disponibilidad muy altas, como así también el incremento del poder de computación.

El diseño modular proporciona una flexibilidad importante y facilita la expansión de la capacidad.

**Detección Automática del Paralelismo**

Los multiprocesadores hacen posible la explotación del paralelismo.

Los sistemas de computación obtienen los beneficios del procesamiento concurrente más por la *“multiprogramación”* de varios procesos y menos por la explotación del *“paralelismo”* dentro de un solo proceso.

La detección del paralelismo es un problema complejo y la puede efectuar el programador, el traductor del lenguaje, el hardware o el Sistema Operativo.

El paralelismo dentro de los programas puede ser *“explícito”* o *“implícito”.*

Las principales características del *paralelismo explícito* son las que se detallan a continuación:

* Es indicado de forma específica por un programador mediante una “construcción de concurrencia” como la siguiente:

**cobegin;**

**proposición 1;**   
**................**   
**proposición n;**

**coend;**

* Se pueden utilizar procesadores separados para ejecutar cada una de las proposiciones.
* Es susceptible de errores de programación difíciles de detectar y depurar.
* El programador puede omitir tratar situaciones donde sería aplicable el paralelismo.

Las principales características del *paralelismo implícito* son las que se detallan a continuación:

* La verdadera esperanza está en la detección automática del paralelismo implícito.
* Es el paralelismo intrínseco del algoritmo pero no establecido explícitamente por el programador.
* Los compiladores explotan el paralelismo implícito mediante las técnicas de “distribución de ciclos” y de “reducción de la altura del árbol”.

**Distribución de Ciclos**

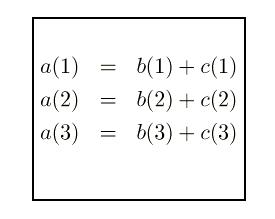
Una “estructura de ciclos o de repetición” implica la repetición de una serie de proposiciones (cuerpo del ciclo) hasta que ocurre alguna condición de terminación, por ejemplo:

**For *i* = 1 to 3**   
**Do**



**;**

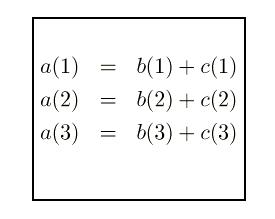
El procesador secuencial realizará en secuencia lo siguiente:



En un sistema de multiprocesamiento con tres procesadores disponibles se podrían ejecutar concurrentemente.

Un compilador que detecte automáticamente el paralelismo implícito puede convertir el ciclo del ejemplo 2.2 en lo siguiente:

**cobegin;**



**coend;**

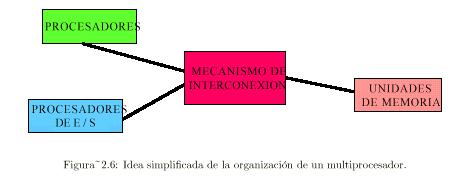
Esta técnica se denomina **distribución de ciclos.**

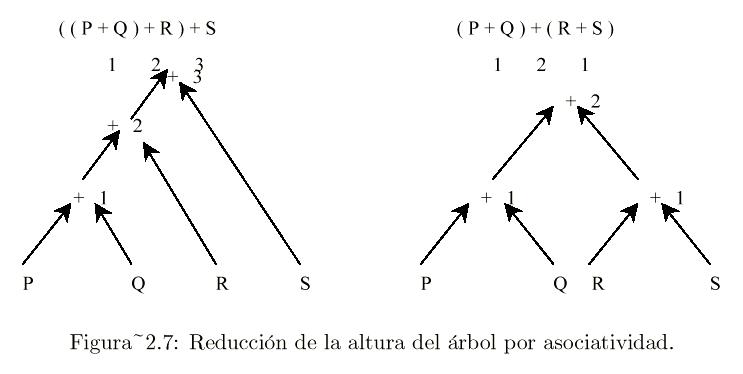
**Reducción de la Altura del Arbol**

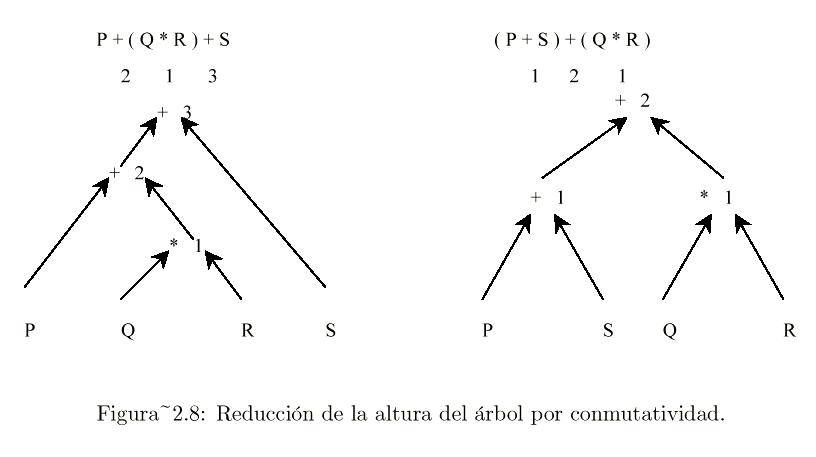
Utilizando las propiedades asociativa, conmutativa y distributiva de la aritmética, los compiladores pueden:

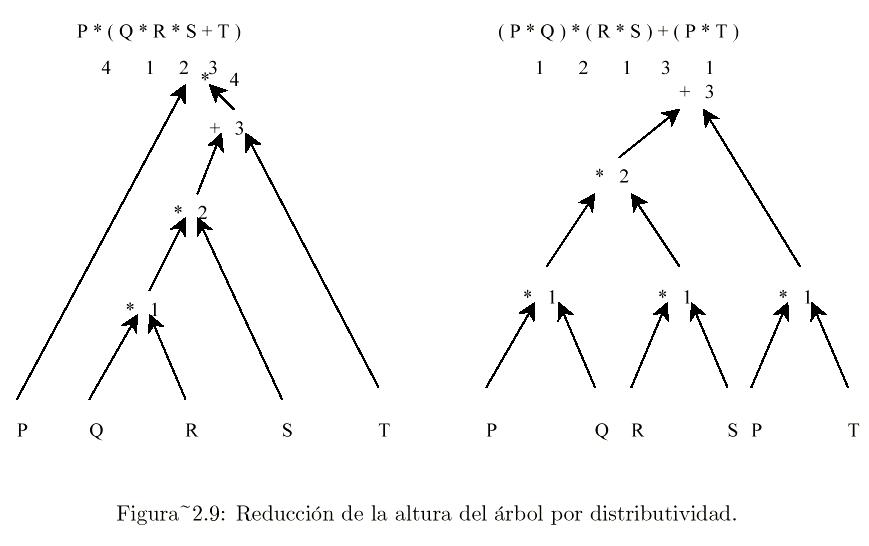
1. Detectar el paralelismo implícito en expresiones algebraicas.
2. Producir un código objeto para multiprocesadores que indique las operaciones que se pueden realizar simultáneamente.
3. Reordenar expresiones para que sean más apropiadas para la computación en paralelo.

Se invierten más tiempo y recursos durante la compilación para reducir el tiempo de ejecución, es decir que se busca optimización en el momento de la compilación para lograr ejecución en tiempo mínimo, lo que es aplicable especialmente cuando los sistemas pasan a producción, no tanto cuando están en desarrollo (ver Figura 2.6, Figura 2.7, Figura 2.8 y Figura 2.9 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)).









*REGLA DE “NUNCA ESPERAR”: Es mejor darle a un procesador una tarea que puede llegar a no ser utilizada, que tenerlo ocioso.*

**Organización del Hardware del Multiprocesador**

El problema clave es determinar los medios de conexión de los procesadores múltiples y los procesadores de Entrada / Salida a las unidades de almacenamiento [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7).

Los multiprocesadores se caracterizan por los siguientes aspectos:

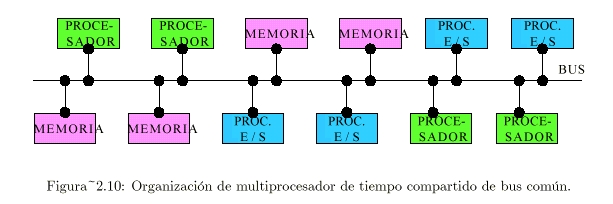
* Un multiprocesador contiene dos o más procesadores con capacidades aproximadamente comparables.
* Todos los procesadores comparten el acceso a un almacenamiento común y a canales de Entrada / Salida, unidades de control y dispositivos.
* Todo está controlado por un Sistema Operativo que proporciona interacción entre procesadores y sus programas en los niveles de trabajo, tarea, paso, archivo y elementos de datos.

Las organizaciones más comunes son las siguientes:

* Tiempo compartido o bus común (conductor común).
* Matriz de barras cruzadas e interruptores.
* Almacenamiento de interconexión múltiple.

**Tiempo Compartido o Bus Común (o Conductor Común)**

Usa un solo camino de comunicación entre todas las unidades funcionales (ver Figura 2.10 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)).



El bus común es en esencia una unidad pasiva.

Un procesador o procesador de Entrada / Salida que desee transferir datos debe efectuar los siguientes pasos:

1. Verificar la disponibilidad del conductor y de la unidad de destino.
2. Informar a la unidad de destino de lo que se va a hacer con los datos.
3. Iniciar la transferencia de datos.

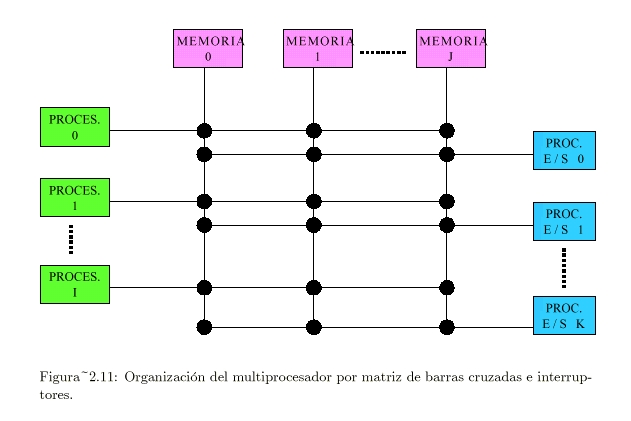
Las unidades receptoras deben poder reconocer qué mensajes del bus son enviados hacia ellas y seguir y confirmar las señales de control recibidas de la unidad emisora.

Es una organización económica, simple y flexible pero con una sola vía de comunicación, por lo cual:

* El sistema falla totalmente si falla el bus.
* La tasa neta de transmisiones está limitada por la tasa neta de transmisión del conductor.
* La contención por el uso del bus en un sistema sobrecargado puede ocasionar una seria degradación.

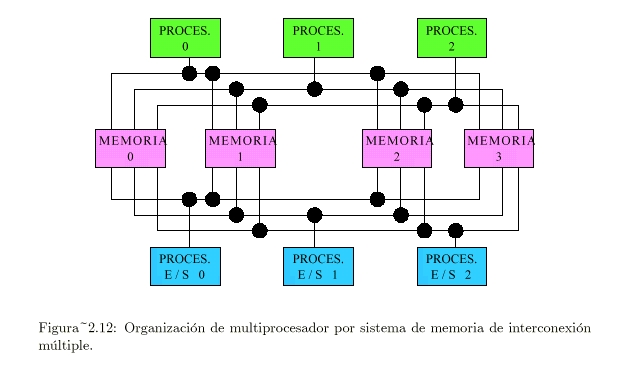
**Matriz de Barras Cruzadas e Interruptores**

En este caso existe un camino diferente para cada unidad de almacenamiento, por lo cual las referencias a dos unidades diferentes de almacenamiento no son bloqueantes sino simultáneas y la multiplicidad de caminos de transmisión puede proporcionar tasas de transferencia muy altas (ver Figura 2.11 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)).



**Almacenamiento de Interconexión Múltiple**

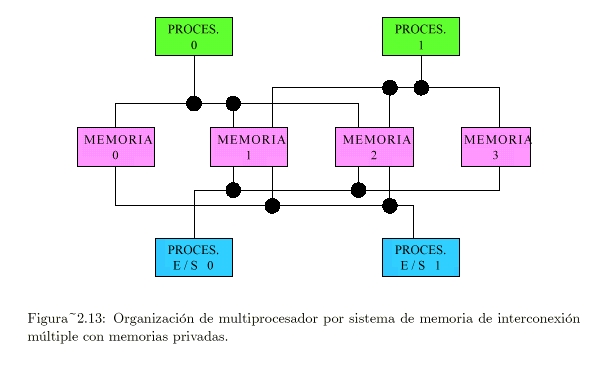
Se obtiene al sacar las lógicas de control, de conmutación y de arbitraje de prioridades fuera del interruptor de barras cruzadas y se las coloca en la interfaz de cada unidad de almacenamiento (ver Figura 2.12 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)).



Cada unidad funcional puede acceder a cada unidad de almacenamiento, pero sólo en una “conexión de almacenamiento” específica, es decir que hay una conexión de almacenamiento por unidad funcional.

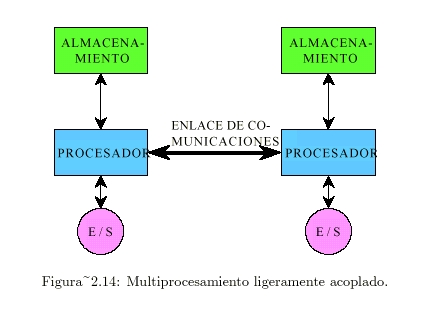
El conexionado es más complejo que en los otros esquemas.

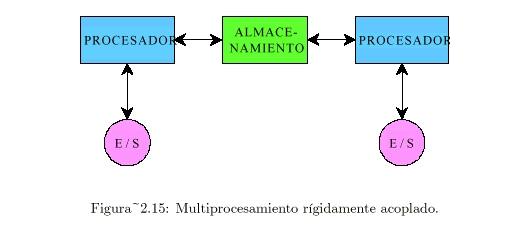
Se puede restringir el acceso a las unidades de almacenamiento para que no todas las unidades de procesamiento las accedan, en tal caso habrá unidades de almacenamiento “privadas” de determinados procesadores (ver Figura 2.13 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)).



**Grados de Acoplamiento en Multiprocesamiento**

Los grados de acoplamiento en multiprocesamiento pueden clasificarse de **ligeramente acoplados** (ver Figura 2.14 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)) o **rígidamente acoplados** (ver Figura 2.15 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)), según las características que se detallan en la Tabla 2.5 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7).





|  |  |
| --- | --- |
| **Grados de acoplamiento en multiprocesamiento** | **Descripción** |
| Ligeramente acoplado | Incluye la conexión de dos o más sistemas independientes por medio de un enlace de comunicación.  Cada sistema tiene su propio Sistema Operativo y almacenamiento.  Los sistemas pueden funcionar independientemente y se comunican cuando sea necesario.  Los sistemas separados pueden acceder a los archivos de los otros e intercambiar tareas a procesadores menos cargados. |
| Rígidamente acoplado | Utiliza un solo almacenamiento compartido por varios procesadores.  Emplea un solo Sistema Operativo que controla todos los procesadores y el hardware del sistema. |
| **Tabla 2.5: Grados de acoplamiento en multiprocesamiento.** | |

**Organización Maestro / Satélite**

Un procesador está diseñado como el “maestro” y los otros como “satélites”.

El procesador “maestro” es de propósito general y realiza operaciones de Entrada / Salida y computaciones.

Los procesadores “satélites” sólo realizan computaciones.

Los procesos limitados por computación pueden ejecutarse con efectividad en los satélites.

Los procesos limitados por la Entrada / Salida ejecutados en los satélites generan frecuentes llamadas de servicios al procesador maestro, pudiendo resultar ineficientes.

Si falla un satélite se pierde capacidad computacional pero el sistema no falla.

Si falla el maestro el sistema falla al no poder efectuar operaciones de Entrada / Salida, por lo que un satélite debería asumir las funciones del maestro previo cambio de los periféricos y reinicio del sistema.

En el multiprocesamiento simétrico todos pueden hacer Entrada / Salida.

**Sistema Operativo de Multiprocesadores**

Las capacidades funcionales de los Sistema Operativo de multiprogramación y de multiprocesadores incluyen lo siguiente:

* Asignación y administración de recursos.
* Protección de tablas y conjuntos de datos.
* Prevención contra el interbloqueo del sistema.
* Terminación anormal.
* Equilibrio de cargas de Entrada / Salida.
* Equilibrio de carga del procesador.
* Reconfiguración.

Las tres últimas son especialmente importantes en Sistemas Operativos de multiprocesadores, donde es fundamental explotar el paralelismo en el hardware y en los programas y hacerlo automáticamente.

Las organizaciones básicas de los Sistemas Operativos para multiprocesadores son las siguientes:

* Maestro / satélite.
* Ejecutivo separado para cada procesador.
* Tratamiento simétrico (o anónimo) para todos los procesadores.

**Maestro / Satélite**

Es la organización más fácil de implementar.

No logra la utilización óptima del hardware dado que sólo el procesador maestro puede ejecutar el Sistema Operativo y el procesador satélite sólo puede ejecutar programas del usuario.

Las interrupciones generadas por los procesos en ejecución en los procesadores satélites que precisan atención del Sistema Operativo deben ser atendidas por el procesador maestro y por ello pueden generarse largas colas de requerimientos pendientes.

**Ejecutivos Separados**

Cada procesador tiene su propio Sistema Operativo y responde a interrupciones de los usuarios que operan en ese procesador.

Existen tablas de control con información global de todo el sistema (por ejemplo, lista de procesadores conocidos por el Sistema Operativo) a las que se debe acceder utilizando exclusión mutua.

Es más confiable que la organización maestro / satélite.

Cada procesador controla sus propios recursos dedicados.

La reconfiguración de los dispositivos de Entrada / Salida puede implicar el cambio de dispositivos a diferentes procesadores con distintos Sistemas Operativos.

La contención sobre las tablas del Sistema Operativo es mínima.

Los procesadores no cooperan en la ejecución de un proceso individual, que habrá sido asignado a uno de ellos.

**Tratamiento Simétrico**

Es la organización más complicada de implementar y también la más poderosa y confiable.

El Sistema Operativo administra un grupo de procesadores idénticos, donde cualquiera puede utilizar cualquier dispositivo de Entrada / Salida y cualquiera puede referenciar a cualquier unidad de almacenamiento.

El Sistema Operativo precisa código reentrante y exclusión mutua.

Es posible equilibrar la carga de trabajo más precisamente que en las otras organizaciones.

Adquieren significativa importancia el hardware y el software para resolución de conflictos.

Todos los procesadores pueden cooperar en la ejecución de un proceso determinado.

El **procesador ejecutivo** es el responsable (uno sólo) en un momento dado de las tablas y funciones del sistema; así se evitan los conflictos sobre la información global.

**Rendimiento del Sistema de Multiprocesamiento**

Aún con multiprocesamiento completamente simétrico, la adición de un nuevo procesador no hará que la capacidad de ejecución del sistema aumente según la capacidad del nuevo procesador, siendo las causas las siguientes:

* Hay sobrecarga adicional del Sistema Operativo.
* Se incrementa la contención por recursos del sistema.
* Hay retrasos del hardware en el intercambio y en el encaminamiento de las transmisiones entre un número mayor de componentes.

Al incrementar el número de procesadores *“n”* similares en un multiprocesador, el incremento de la productividad no es lineal y tiende a disminuir cuando *“n”* crece.

**Recuperación de Errores**

Una de las capacidades más importantes de los Sistemas Operativos de multiprocesadores es la de soportar fallas de hardware en procesadores individuales y continuar su operación.

Debe existir el soporte correspondiente en el Sistema Operativo.

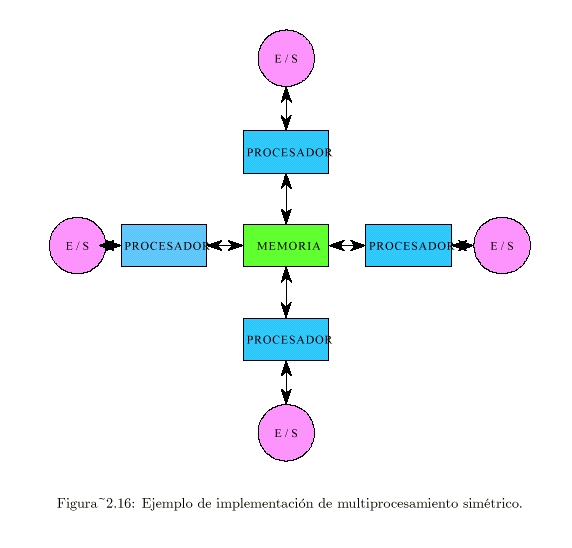
Las técnicas de recuperación de errores incluyen los siguientes aspectos:

* Los datos críticos (del sistema y de usuario) deben mantenerse en copias múltiples y en bancos de almacenamiento separados.
* El Sistema Operativo debe ejecutar efectivamente con la configuración máxima y con subconjuntos ante fallas.
* Debe haber capacidad de detección y corrección de errores de hardware sin interferir con la eficiencia operacional del sistema.
* Se debe utilizar la capacidad ociosa del procesador para tratar de detectar posibles fallos antes de que se produzcan.
* El Sistema Operativo debe dirigir un procesador operativo para que tome el control de un proceso que se estaba ejecutando en un procesador que falla.

**Multiprocesamiento Simétrico (MPS)**

Cada procesador posee capacidades funcionales completas.

Los dispositivos de Entrada / Salida pueden ser conectados a cada uno de los procesadores (ver Figura 2.16 [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7)).



Todas las llamadas al supervisor pueden ser ejecutadas en todos los procesadores, inclusive las de Entrada / Salida.

Si un programa en ejecución en un procesador pide una operación de Entrada / Salida en un dispositivo conectado a un procesador diferente, el procesador puede continuar ejecutando el trabajo y la Entrada / Salida se coloca en una cola para su iniciación por el procesador apropiado.

Se considera **procesador ejecutante** al que está ejecutando un proceso determinado.

Se considera **procesador propietario** al que está conectado a los diferentes dispositivos utilizados por el proceso.

Es más eficiente que la organización maestro / satélite, ya que los requerimientos de Entrada / Salida se encolan y no sobrecargan con intercambio de contexto y a que en la organización maestro / satélite las peticiones de Entrada / Salida en el satélite provocan un intercambio de contexto en el maestro.

Cada procesador puede ejecutar el planificador para buscar el siguiente trabajo a ejecutar, de forma que un proceso determinado se ejecuta en diferentes procesadores en distintos momentos; además, el MPS utiliza una sola cola de trabajos y cada procesador puede seleccionar trabajos de ella, con lo cual se equilibra la carga entre los procesadores.

Para minimizar la contención en el despacho de procesos, los relojes de los procesadores tienen oblicuidad, debido a ello las interrupciones de reloj ocurren en diferentes momentos.

**Tendencias de los Multiprocesadores**

Todo indica que el uso de los multiprocesadores se incrementará considerablemente en el futuro [[7, Deitel]](http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/SOB.htm#7).

Las principales razones son las siguientes:

* La confiabilidad requerida es cada vez mayor.
* La reducción de costos consecuencia de los avances en microelectrónica.
* El previsible desarrollo de lenguajes que permitan a los usuarios expresar el paralelismo explícitamente.
* El progreso en la detección automática del paralelismo .
* El hecho de que se estaría llegando a los límites del uniprocesador debido a la compactación de componentes, es decir que se estaría próximo a los límites de longitud y de proximidad de los “caminos electromagnéticos” (longitud del recorrido de la señal electromagnética); alcanzados los límites mencionados, la única posibilidad de incrementar capacidad de cómputo es mediante multiprocesamiento.

Existen estudios de tendencias en arquitectura de computadoras que apuntan a los **poliprocesadores**, es decir, sistemas que combinan el multiprocesamiento, simétrico y asimétrico, para crear una jerarquía de procesadores dentro de un sistema.