cg_enri
www.wuolah.com/student/cg_enri

tema2introduccionSO.pdf

Tema 2 Introducción SO

1° Fundamentos del Software

Grado en Ingeniería Informática

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Tema 2: Introducción a los SO

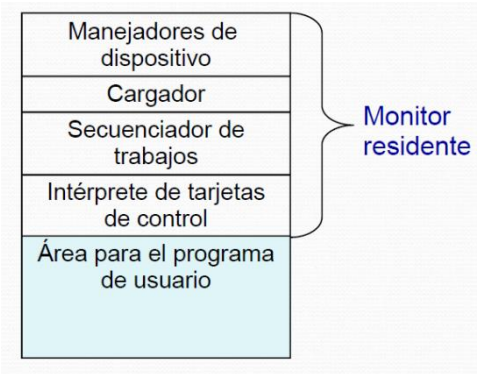
Recordemos antes de nada la definición de SO:

Un sistema operativo es en sí un programa que funciona como interfaz entre las aplicaciones y el hardware del computador. Este persigue los siguientes objetivos:

- Facilitar el uso: actúa de interfaz entre el usuario final y el computador, el usuario no debe de preocuparse por los detalles. También facilitan la tarea del programador ya que el S unido a los lenguajes de programación de alto nivel, hacen que le programador no se preocupa por el lenguaje máquina.
- Eficiencia: el SO controla los recursos del computador, responsables del transporte, almacenamiento y la ejecución de datos. Por lo el SO es un instrumento de control, que no es externo al propio computador que controla.
- Capacidad de evolucionar: tiene la capacidad de adaptarse a nuevos dispositivos hardware, mejorar y solucionar errores detectados.

Evolución del sistema operativo

- **Procesamiento en serie:** como no existe un SO, todas las tareas las realiza el programador en lenguaje máquina necesitando altos niveles de planificación y utilizando pocos recursos de la CPU. Se introducían las instrucciones en tarjetas y en caso de error, el usuario detectaba la causa. Había que cambiar el proceso manualmente.
  Resumen:
  - No existe SO → el usuario interactúa con la máquina directamente
  - Problemas → baja utilización del tiempo de la CPU. Alto tiempo de planificación
- **Sistemas por lotes (Sistemas Batch):**
  Permite la utilización de un monitor. Un elemento de software que permite que el programador no tenga que acceder al lenguaje máquina. De este modo, el programador introducía una tarjeta con el trabajo a realizar, a partir de la cual se creaba un sistema de lotes con todos los trabajos enviados. Cuando finalizaba un trabajo, el control pasaba al monitor y este ejecutaba el siguiente. El resultado de cada trabajo se enviaba al correspondiente dispositivo de salida. Por lo que la función de planificación no lo hacía hacia el usuario, el monitor realizaba un trabajo eficiente tras otro.



  Resumen:
  - Agrupa trabajos similares y reduce el tiempo de planificación
  - Trabajo = (programa + datos + órdenes de control para el sistema)

INEAF
BUSINESS SCHOOL

Es el momento
DE CRECER

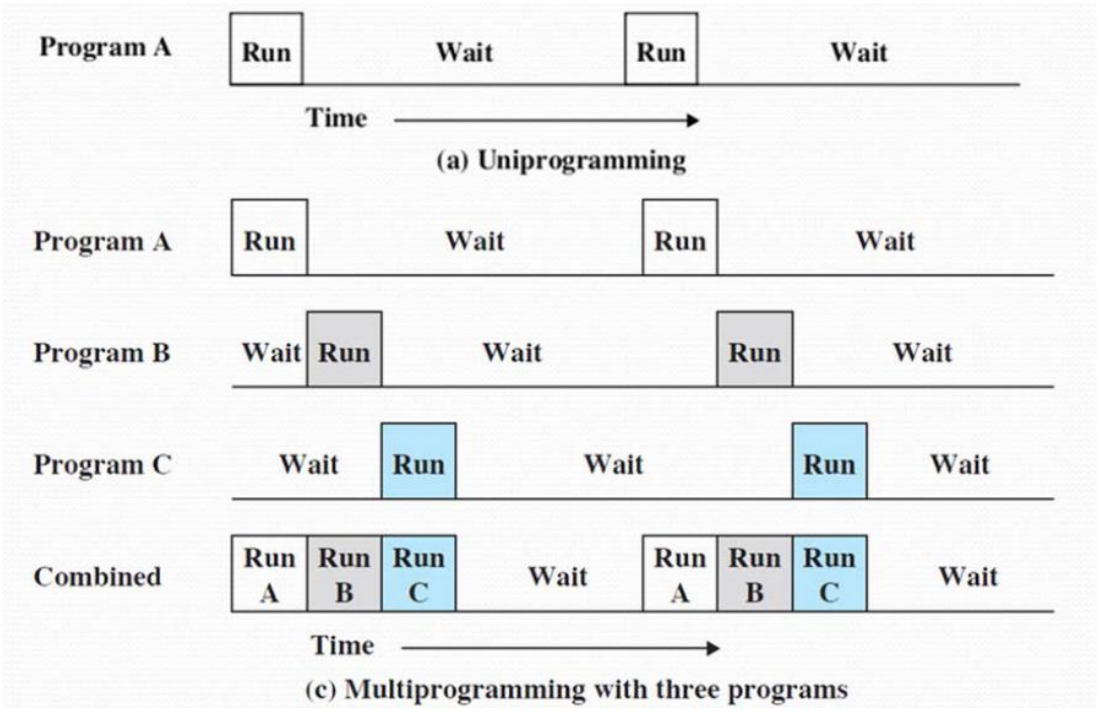
Master en Asesoría Fiscal de Empresas

- Falta de interacción entre le usuario y el computador mientras se ejecuta el trabajo
- Problemas: CPU ociosa durante E/S → Solución → Spooling, superpone la E/S de un trabajo al cómputo de otros.

Sistemas multiprogramados

En el sistema por lotes se mejoraba respecto a los sistemas anteriores ya que cada programa se ejecutaba después del anterior, pero había que esperar a que el anterior terminara.

En un sistema multiprogramado no hay que esperar a la finalización de un proceso para la ejecución de otro. Cuenta con varios procesadores que trabajan al mismo tiempo (multiprocesador) y aprovecha los espacios de tiempo entre medias para ejecutar otro programa. Optimizando el tiempo de ejecución y los recursos del CPU.



Multiprogramación con 3 programas

Definiciones

- **SO multiprogramado:** SO que permite la ejecución de más de un proceso simultáneamente. → Sus datos e instrucciones están en memoria principal.
- **SO monousuario:** proporciona servicios a un usuario.
- **SO multiusuario:** proporciona servicios a varios usuarios.
- **SO monoprocesador:** gestiona un sistema de computación con un procesador.

- **SO multiprocesador:** gestiona un sistema de computación con varios procesadores.
- **SO de tiempo compartido:** SO multiprogramado donde se realiza un reparto del tiempo del procesador en pequeños trozos de tal forma que todos los procesos pueden avanzar adecuadamente.

Procesos

Definición de proceso. Hay varias:

- Un programa en ejecución.
- Una instancia de un programa ejecutándose en un ordenador.
- La entidad que se puede asignar o ejecutar en un procesador.
- Una unidad de actividad caracterizada por un solo flujo de ejecución, un estado actual y un conjunto de recursos del sistema asociados.

Está formado por:

- Un programa ejecutable.
- Datos que necesita el SO para ejecutar el programa.
- El contexto de ejecución del programa o el estado del proceso, que es el conjunto de datos internos por el cual el SO es capaz de supervisar y controlar el proceso.

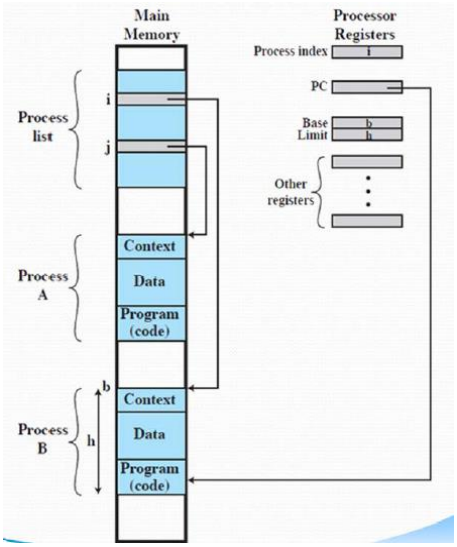
Bloque de control de procesos (PCB, Process Control Block)

Para que los datos de un proceso queden guardados a la hora de sustituirlo por otro, usamos el PCB. En él se guarda la información pertinente para posteriormente ejecutar el programa desde el mismo punto. Entre esta información encontramos:

- Identificador: asocia un identificador único al proceso para distinguirlo.
- Estados: guarda el estado actual del programa.
- Prioridad: se establece una jerarquía para que los procesos se ejecuten de forma ordenada y repartida.
- Punteros a memoria: se localizan los punteros al código de programa necesario para su ejecución.
- Datos de contexto: datos que están presentes en los registros del procesador cuando el proceso se está ejecutando.
- Información de estado de E/S: incluye las peticiones a E/S realizadas, así como los dispositivos que van a ser utilizados.
- Información de auditoría: tiempo utilizado por el procesador, etc.

Implementación de procesos típica

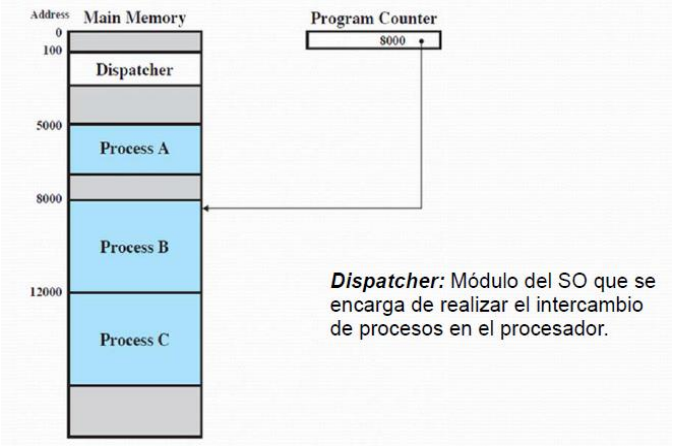
- Esta implementación permite en el proceso como una estructura de datos.
- El estado completo del proceso en un instante donde se almacena en su PCB.
- Esta estructura permite el desarrollo de técnicas potentes que aseguren la coordinación y la cooperación entre los procesos.



Concepto de traza de ejecución

Es un listado de la secuencia de las instrucciones de un programa que realiza el procesador para un proceso.

Desde el punto de vista del procesador, se entremezclan las trazas de ejecución de los procesos y las trazas de código del SO.

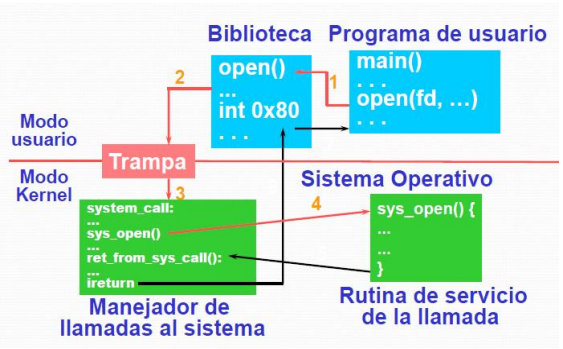


Llamadas al sistema

Es la forma en la que se comunican los programas del usuario con el SO, en tiempo de ejecución. Son peticiones de servicios que se hacen en el proceso al SO,

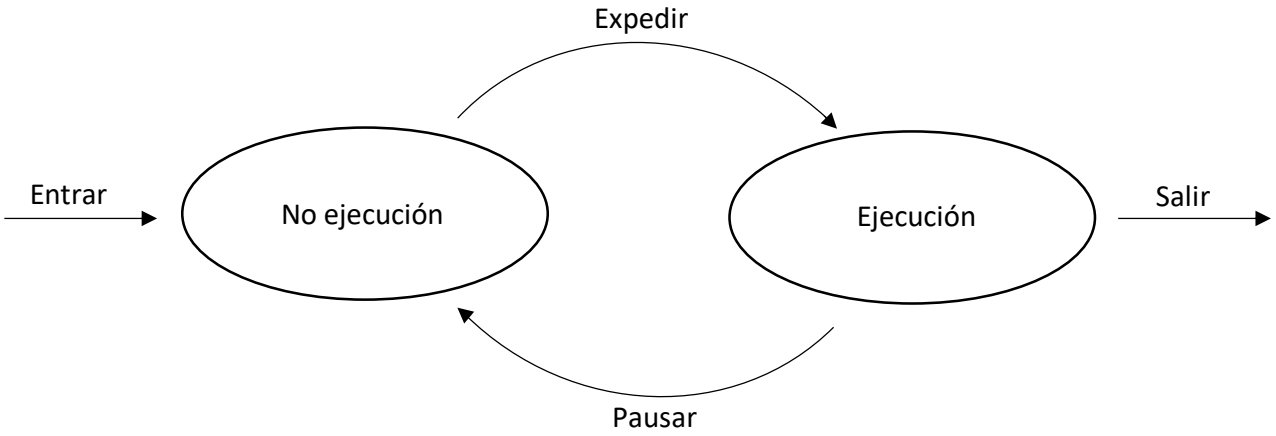
Ejemplos de llamadas al sistema:

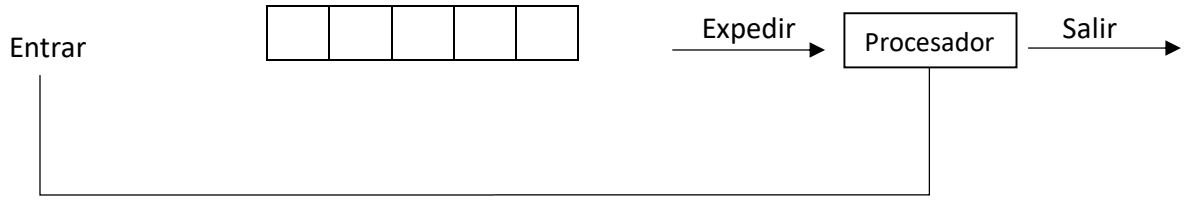
- Solicitudes de E/S
- Gestión de procesos
- Gestión de memoria



Se implementa a través de una trampa (trap) o "interrupción software".

Modelo de 2 estados → Estados de un proceso



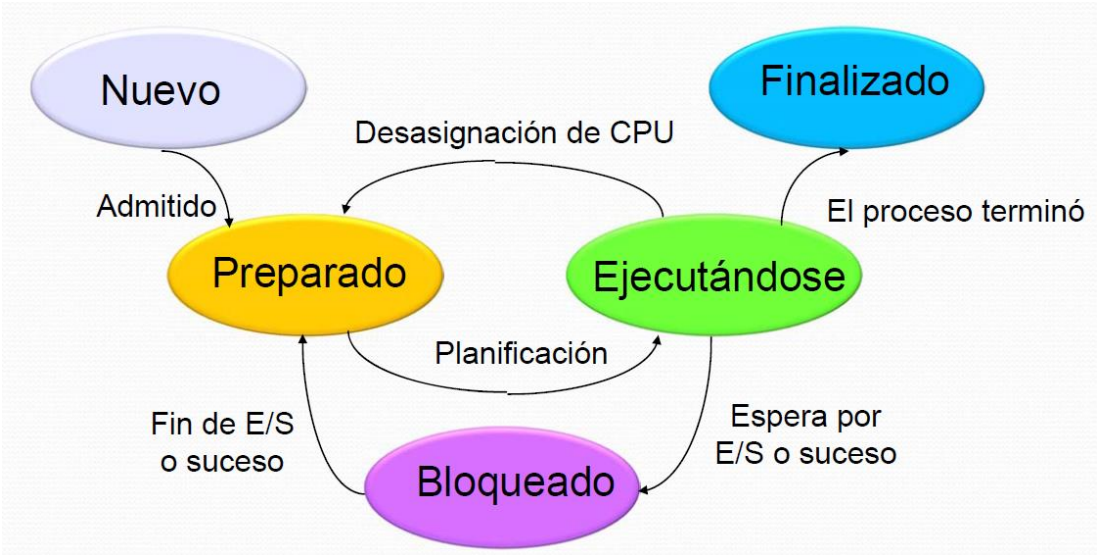


Modelo de los 5 estados

Este modelo trata de representar las actividades que el SO lleva a cabo sobre los procesos:

- **Creación:** cuando se va a añadir un nuevo proceso, el SO construye estructuras de datos que usa para mejorar el proceso y reservar el espacio de direcciones en memoria principal.
- **Terminación:** todo sistema debe proporcionar los mecanismos mediante los cuales un proceso indica su finalización. Esta acción tiene como resultado una solicitud de un servicio al SO para terminar, un nº de error o una condición de fallo pueden llevar a la finalización de un proceso.
- **Multiprogramación:** Para ello se hace uso del modelo de los 5 estados:
  - **Nuevo:** un proceso que se acaba de crear u que aún no ha sido admitido en el grupo de procesos ejecutables por el SO. No se ha cargado en memoria principal, pero su bloque de control de procesos sí ha sido creado.
  - **Preparado o listo:** un proceso preparado para ejecutarse cuando tenga oportunidad.

- **Ejecutándose:** si el ordenador solo tiene 1 procesador, solo un proceso se puede estar ejecutando.
- **Bloqueado:** proceso que no se puede ejecutar hasta que se cumpla un determinado evento o una operación de E/S.
- **Finalizado o saliente:** proceso que ha sido liberado del grupo de procesos ejecutables por el SO.



Transiciones entre estados

- Nuevo → Preparado: PCB está creado y el programa disponible en memoria.
- Ejecutándose → Finalizado: el proceso finaliza anormalmente o es abortado por el SO a causa de un error.
- Preparado → Ejecutándose: SO selecciona un proceso para que se ejecute en el procesador.
- Ejecutándose → Bloqueado: el proceso solicita algo al SO, hay que esperar.
- Ejecutándose → Preparado: un proceso ha alcanzado un máximo de tiempo de ejecución ininterrumpida.
- Bloqueado → Preparado: se produce el evento por el cual el SO bloqueó al proceso.
- Preparado o Bloqueado → Finalizado: finalización de un proceso por parte de otro → no se suele permitir.

Descripción y control de procesos

Descripción de procesos: PCB

- Identificadores: del proceso, su parche, el usuario.
- Contexto de ejecución: valores de registros PC, PSW, SP.
- Información para control de proceso:
  - Información de estado y planificación
  - Descripción de las regiones de memoria asignadas

- Recursos asignados
- Enlaces a colas de procesos
- Comunicación entre procesos

Creación de un proceso: inicialización de PCB

- Asignar un identificador único al proceso.
- Asignar un nuevo PCB.
- Asignar memoria para el programa asociado.
- Inicializar PCB:
  - PC: dirección inicial de comienzo del programa
  - SD: dirección pila del sistema
  - Memoria: donde reside el programa
  - El resto de campos se inicializa a valores por omisión

Control de estados: modos de ejecución

- **Modo usuario:** El programa ejecutado en este modo tiene:
  - Un subconjunto de los registros del procesador
  - Un subconjunto de repertorio de instrucciones máquina
  - Un área de memoria
- **Modo núcleo (kernel, supervisor o sistema):** El programa (SO) que se ejecuta en este modo tiene acceso a todos los recursos de la máquina, software y hardware.

¿Cómo utilizar el SO en modo de ejecución?

- El modo de ejecución (incluido en el PSW) cambia a modo kernel, automáticamente por hardware, cuando se produce:
  - + Una interrupción
  - + Una excepción
  - + Una llamada al sistema

  Cambio de modo usuario a modo Kernel

  No implica cambio contexto
- Seguidamente se ejecuta la rutina del SO correspondiente al cuento producido.
- Finalmente, cuando termina la rutina, el hardware restaura automáticamente el modo de ejecución al modo usuario

Un cambio de modo **NO IMPLICA** un cambio de contexto

Control de procesos: cambio de modo

- Se ejecuta una rutina del SO en el contexto del proceso que se encuentra en estado “ejecutándose”.

- ¿Cuándo puede realizarse?. Siempre que el SO puede ejecutarse luego. solamente como resultado de:
  - + Llamada al sistema
  - + Excepción
  - + Interrupción

Pasos en la operación de cambio de modo

1) El hardware automáticamente salva como mínimo PC y PSW y cambia el bit de modo usuario a modo kernel.
2) Determina automáticamente la ruta del SO que debe ejecutarse y cargar el PC con su dirección de comienzo.
3) Ejecutar la rutina. Posiblemente la rutina comience salvando el resto de registros del procesador y termine restaurando en el procesador la información de registros previamente salvada.
4) Volver a la rutina del SO al proceso que se está ejecutando. El hardware automáticamente restaura en el procesador la información del PC y PSW previamente salvada.

Control de proceso: cambio de contexto

- Un proceso en estado “ejecutándose” cambia a otro estado y un proceso en estado “preparado” pasa a estado “ejecutándose”.
- Se puede realizar cuando el SO puede ejecutarse y decide llevarlo a cabo luego, solamente como resultado de :
  - + Un interrupción
  - + Una excepción
  - + Una llamada al sistema

Un cambio de contexto **IMPLICA** un cambio de modo

Pasos en la operación de cambio de contexto (dispactcher)

1) Salvar los registros del procesador en el PCB del proceso que actualmente está en modo “ejecutándose”.
2) Actualizar el campo estado del proceso al nuevo estado al que pase e insertar el PCB en la cola correspondiente.
3) Seleccionar un nuevo proceso del conjunto de lo que se encuentran en estado “preparándose” (Scheduler o Planificador del CPU)
4) Actualizar el estado de proceso seleccionado a “ejecutándose” y sacarlo de la cola de preparados
5) Cargar los registros del procesador con la información de los registros almacenada en el PCB del proceso seleccionado.

Si se produce → Excepción / Llamada al sistema / Interrupción → Cambio de modo usuario a Kernel

No tiene por qué haber un cambio de contexto

Cambio de modo —NO→ Cambio de contexto

Cambio de contexto —Sí→ Cambio de modo

Hebras (hilos)

Concepto de hebra

- El concepto de proceso (tarea) tiene 2 características diferenciadas que permiten al SO:
  - Controlar la asignación de los recursos necesarios por la ejecución de programas
  - La ejecución del programa asociado al proceso de forma intercalada con otros programas
- El concepto de proceso (tarea) y hebras asociadas se basa en separar estas 2 características:
  - La tarea encargada de soportar todos los datos necesarios (incluida la memoria)
  - Cada una de las hebras permite la ejecución del programa de dorma "independiente" del resto de hebras.

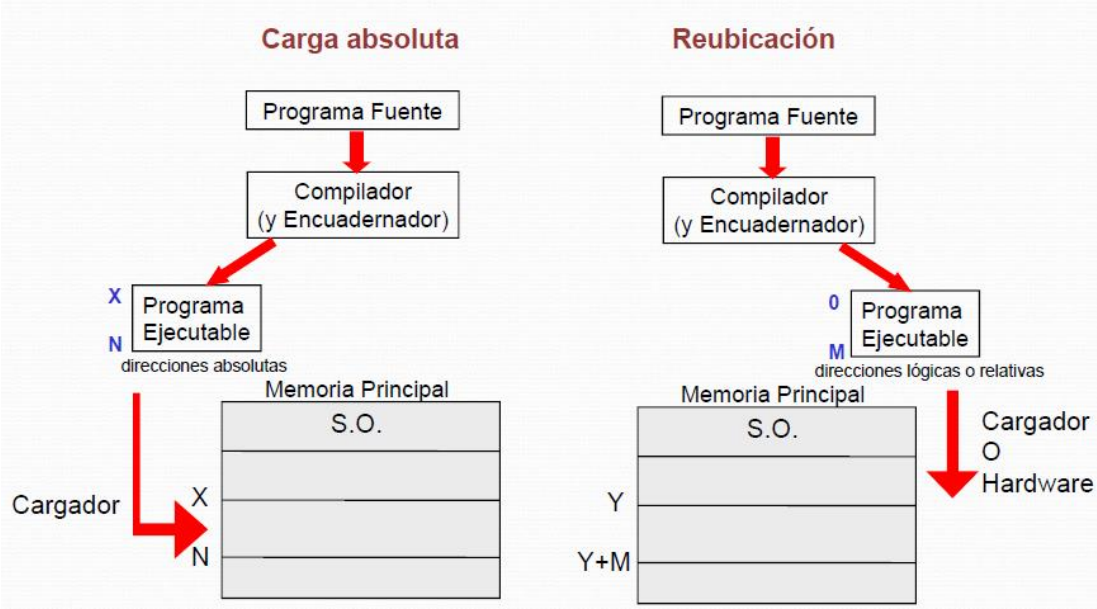
Modelo 5 estados en hebras

Ventajas de las hebras

- Menor tiempo de creación de una hebra en un proceso ya creado que la creación de un nuevo proceso.
- Menor tiempo de finalización de una hebra que de un proceso.
- Menor tiempo de cambio de contexto (hebra) entre hebras pertenecientes al mismo proceso.
- Facilitan la comunicación entre hebras pertenecientes al mismo proceso.
- Permiten aprovechar las técnicas de programación concurrente y el multiprocesamiento simétrico.

Gestión básica de memoria

- **Carga absoluta:** asigna direcciones físicas al programa en tiempo de compilación. El programa no es reubicable.
- **Reubicación:** capacidad ce cargar y ejecutar un programa en un lugar arbitrario en la memoria.



Reubicación estática

- El compilador genera direcciones lógicas (relativas) de 0 a M.
- La decisión de dónde ubicar el programa en memoria principal se realiza en tiempo de carga.
- El cargador añade la dirección base de carga a todas las referencias relativas a memoria del programa.

Reubicación dinámica

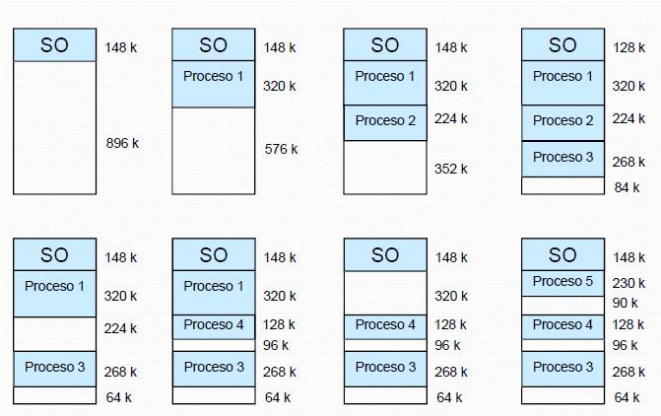
• El compilador genera direcciones lógicas (relativas) de 0 a M.
• La traducción de direcciones lógicas a físicas se realiza en tiempo de ejecución luego el programa está cargado con referencias relativas.
• Requiere apoyo hardware.

Espacios para las direcciones de memoria

- **Espacio de direcciones lógico**: Conjunto de direcciones lógicas (o relativas) que utiliza un programa ejecutable.
- **Espacio de direcciones físico**: Conjunto de direcciones físicas (memoria principal) correspondientes a las direcciones lógicas del programa en un instante dado.
- **Mapa de memoria de un ordenador**: Todo el espacio de memoria direccionable por el ordenador. Normalmente depende del tamaño del bus de direcciones.
- **Mapa de memoria de un proceso:** Se almacena en una estructura de datos (que reside en memoria) donde se guarda el tamaño total del espacio de direcciones lógico y la correspondencia entre las direcciones lógicas y las físicas.

Problema de la fragmentación de memoria

- El modelo comienza correctamente pero finalmente llega a una situación en la cual existen muchos huecos en la memoria.
- A medida que pasa el tiempo, la memoria se fragmenta cada vez más y la memoria se fragmenta cada vez más y la utilización de memoria se decrementa.



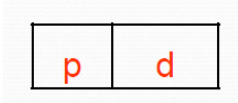
Solución

- "Trocear" el espacio lógico en unidades más pequeñas páginas (elementos de longitud fija) o segmentos (elementos de longitud variable).
- Los trozos no tienen por qué ubicarse consecutivamente en el espacio físico.
- Los esquemas de organización del espacio lógico de direcciones y de traducción de una dirección de espacio lógico al espacio físico que comentamos son:
  - Paginación
  - Segmentación

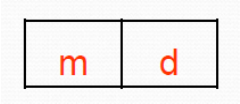
Paginación

- El espacio de dirección física de un proceso puede no ser contiguo.
- La memoria física se divide en bloques de tamaño fijo, denominados "marcos de página".
- El espacio lógico de un proceso se divide en bloques del mismo tamaño, denominadas "páginas".
- Los marcos de página contendrán páginas de los procesos.

Las direcciones lógicas, que son las que genera la CPU, se dividen en número de página (p) y desplazamiento dentro de la página (d).



Las direcciones físicas se dividen en número de marco (m, marco donde está almacenada la página) y desplazamiento (d).



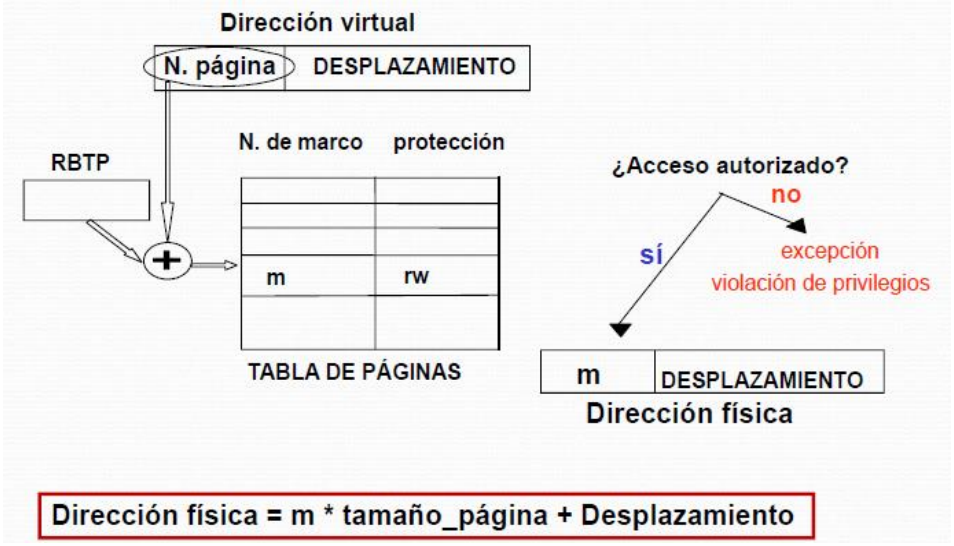
Cuando la CPU genere una dirección lógica será necesario traducirla a la dirección física correspondiente, la tabla de páginas mantiene información necesaria para realizar dicha traducción. Existe una tabla de páginas por proceso.

Tabla de marcos de página, usada por el S.O. y contiene información sobre cada marco de página.

Tablas de páginas

Una entrada por cada página del proceso, contiene:

- Nº de marco: en el que está almacenada la página si está en memoria principal.
- Modo acceso: autorizado a la página (bits de protección).



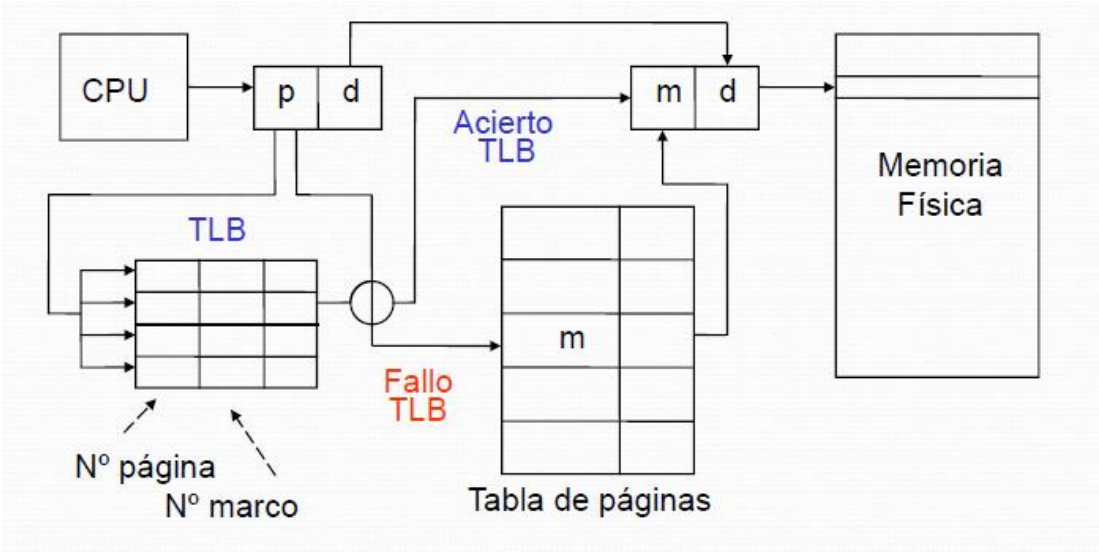
- La tabla de páginas se mantiene en memoria principal.

- El registro base de la tabla de páginas (RBTP) apunta a la tabla de páginas (suele almacenarse en el PCB del proceso).

- En este esquema cada acceso a una instrucción o dato requiere 2 accesos a memoria, uno para la tabla de páginas y otro a memoria.

Búfer de traducción adelantada (TLB)

- El problema de los 2 accesos a memoria se resuelve con un caché hardware de consulta rápida denominada búfer de traducción adelantada o TCB.
- El TCB se implementa como un conjunto de registros asociados que permite una búsqueda en paralelo.
- De esta forma. Para traducir una dirección:
  1. Si existe ya en el registro asociativo, obtenemos el marco.
  2. Si no, la buscamos en la tabla de páginas y se actualiza el TCB, con esta nueva entrada.



Segmentación

Organización de memoria que soporta mejor la visión de memoria del usuario. Un programa es una colección de unidades lógicas, "segmentos".

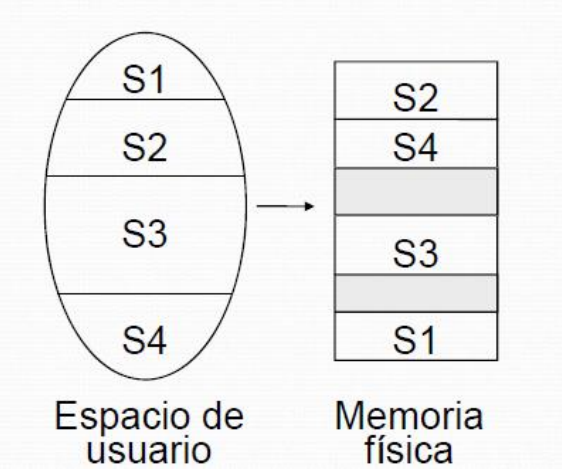


Tabla de segmentos

Una dirección lógica es una tupla: <número_de_segmento, desplazamiento>

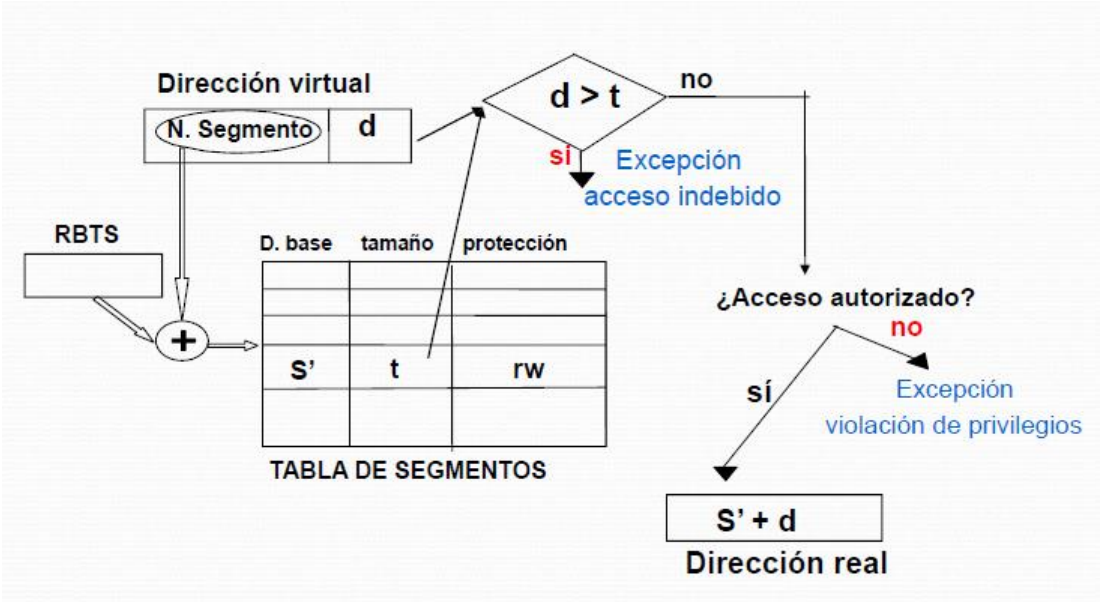
La tabla de segmentos aplica direcciones bidimensionales definidas por el usuario en direcciones físicas a una dimensión. Cada entrada de la tabla tiene los siguientes elementos (a parte presencia, modificación y protección).

- **base** - dirección física donde reside el inicio del segmento en memoria.
- **tamaño** - longitud del segmento

Implementación de la Tabla de Segmentos

- La tabla de segmentos se mantiene en memoria principal.
- El *Registro Base de la Tabla de Segmentos* (*RBTS*) apunta a la tabla de segmentos (suele almacenarse en el PCB del proceso).
- El *Registro Longitud de la Tabla de Segmentos* (*STLR*) indica el número de segmentos del proceso; el nº de segmento *s,* generado en una dirección lógica, es legal si *s* < *STLR* (suele almacenarse en el PCB del proceso).

Esquema de traducción



$$Marcos\ de\ página = \frac{Tamaño\ de\ memoria}{Tamaño\ de\ página}$$