Informática

Unidad 1: Introducción

1B: Sistemas Operativos

Ingeniería en Mecatrónica

Facultad de Ingeniería
Universidad Nacional de Cuyo

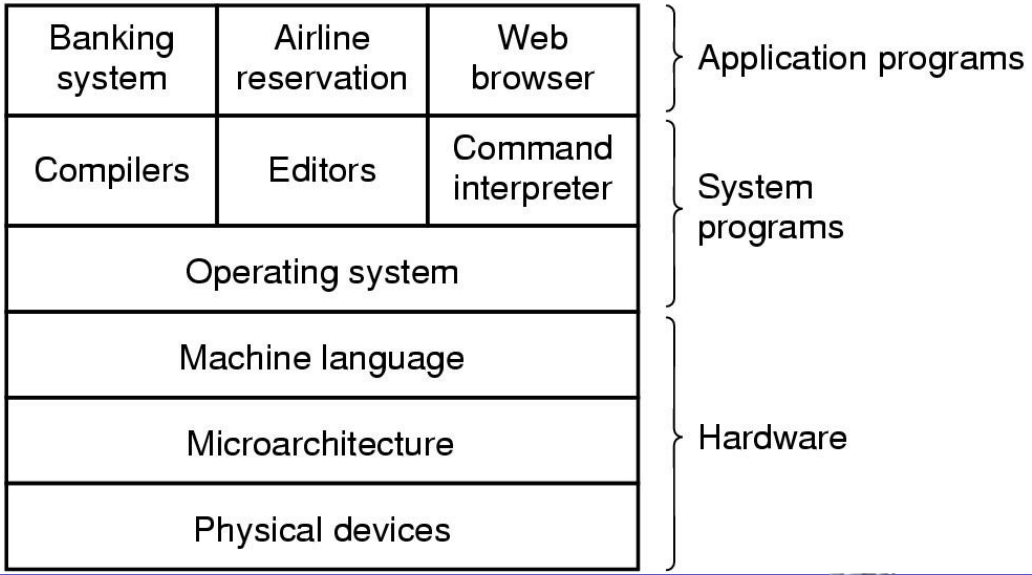
Dr. Ing. Martín G. Marchetta
mmarchetta@fing.uncu.edu.ar

UNIVERSIDAD NACIONAL DE CUYO

FACULTAD DE INGENIERIA
en acción continua...

Unidad 1: Sistemas Operativos

- Una computadora es un sistema complejo
  - Gran variedad de dispositivos: CPU/Cores, memoria, discos, impresoras
  - Distintas propiedades, mecanismos, rendimientos
- Para los programas de usuario, manejar esta complejidad sería ineficiente y costosa
- Se necesita un componente intermedio que oculte esta complejidad a los programas de usuario: el **Sistema Operativo**



Unidad 1: Sistemas Operativos

- Un Sistema Operativo tiene 2 funciones principales
  - Proveer una ***máquina extendida*** o **máquina virtual** al usuario
    - Más fácil de programar que la máquina real, brindando capacidades adicionales (ej: memoria virtual)
  - Ser un controlador de los recursos de cómputo
    - Asignación ordenada de recursos de cómputo (CPU, impresoras, discos, etc)
    - Evitar colisiones, deadlocks, etc. a causa de la concurrencia

Unidad 1: Sistemas Operativos - Conceptos

- La interfaz entre el SO y los programas de usuario es el conjunto de **llamadas al sistema**
- Las llamadas al sistema proporcionan un conjunto de “instrucciones ampliadas”
  - Proveen funcionalidades más complejas que las del hardware, implementadas mediante las funcionalidades más simples que el hardware soporta
- Los componentes más importantes del SO son
  - Procesos
  - Memoria
  - Archivos
  - E/S

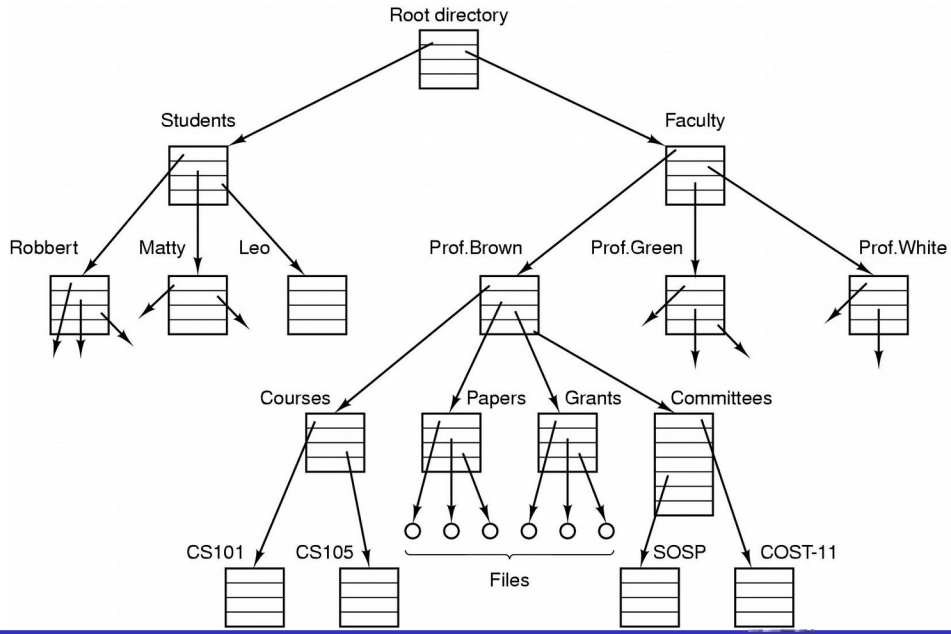
Unidad 1: Sistemas Operativos - Conceptos

- Procesos
  - Un proceso es un programa en ejecución
  - Un proceso está formado por
    - El programa ejecutable (es decir, las instrucciones)
    - Sus registros (pila, puntero de instrucciones, etc).
    - Sus datos (variables y sus valores)
  - El SO mantiene toda esta información para administrar los procesos (suspender/reanudar)
  - Los procesos pueden, a su vez, crear procesos hijos.
    - Ej: Shell
    - La creación/eliminación/suspensión/etc. se realizan mediante llamadas al sistema

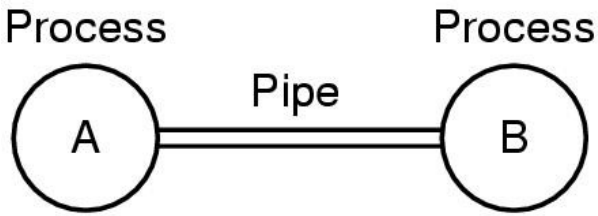
Unidad 1: Sistemas Operativos - Conceptos

- Archivos
  - Simplificación (abstracción) de la realidad de E/S
  - En la mayoría de los sistemas, contienen datos que se leen/escriben en el disco
  - En algunos sistemas, son además una abstracción de otros dispositivos de E/S (ej: dispositivos USB)
  - Operaciones relevantes: creación, eliminación, apertura, cierre, lectura, escritura, búsqueda

- Archivos
  - Los archivos se organizan en un árbol de directorios
  - Cada archivo tiene una **ruta de acceso** (path)
    - Rutas absolutas
    - Rutas relativas
  - En sistemas multi-usuario, los archivos tienen restricciones de acceso
    - Ej: en sistemas Unix se utilizan bits rwx (Read/Write/eXecute)



- Archivos
  - En muchos sistemas existen **archivos especiales**: cuando se lee/escribe en esos archivos en realidad se está accediendo a un dispositvo de E/S
  - 2 tipos:
    - De bloque
    - De carácter
  - Los archivos especiales se protegen con el mismo mecanismo que los archivos regulares (bits rwx)
  - Existe un último tipo de archivo especial, para manejar la comunicación entre procesos: un tubo (o tubería, **pipe** en inglés)

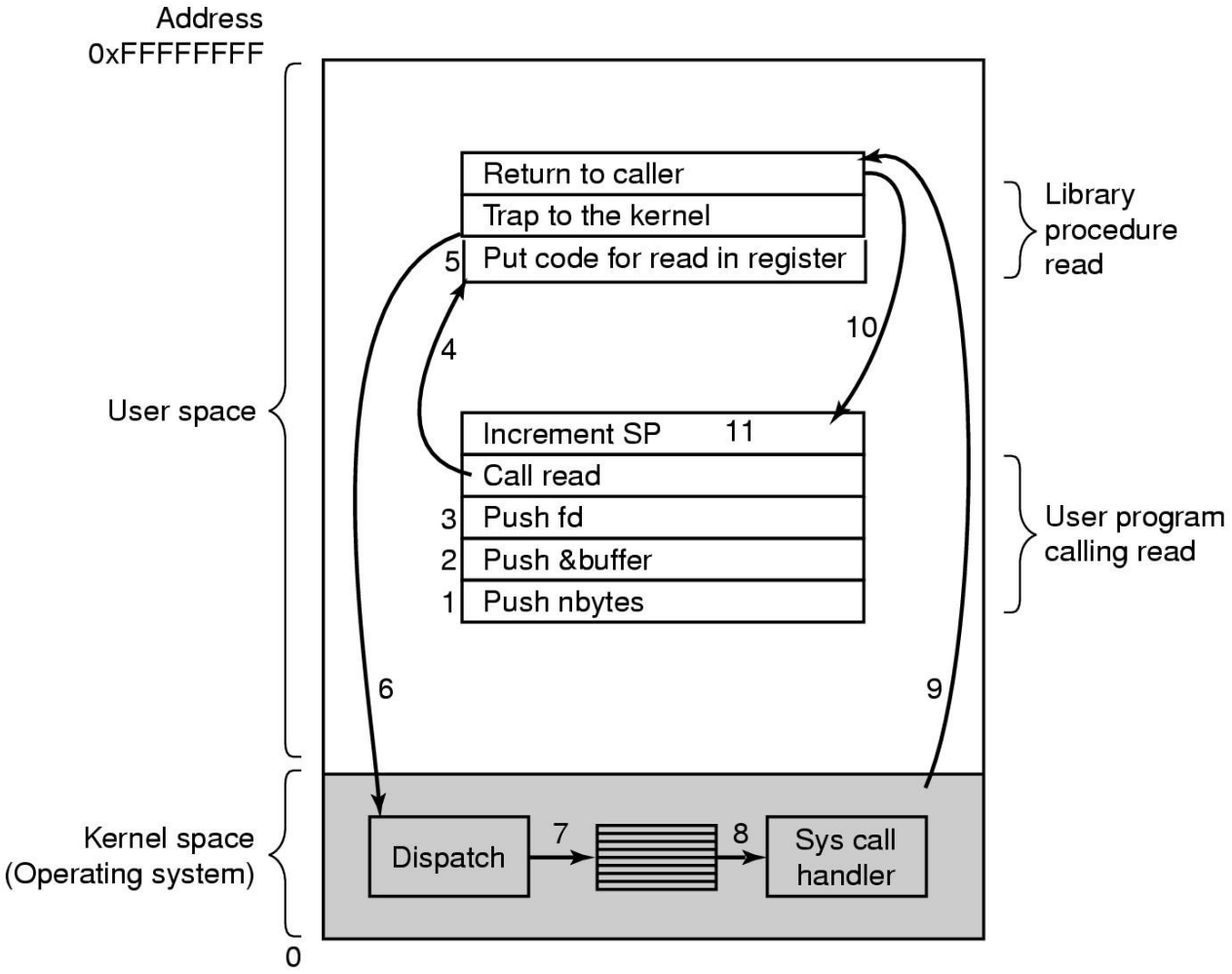


Unidad 1: Sistemas Operativos - Conceptos

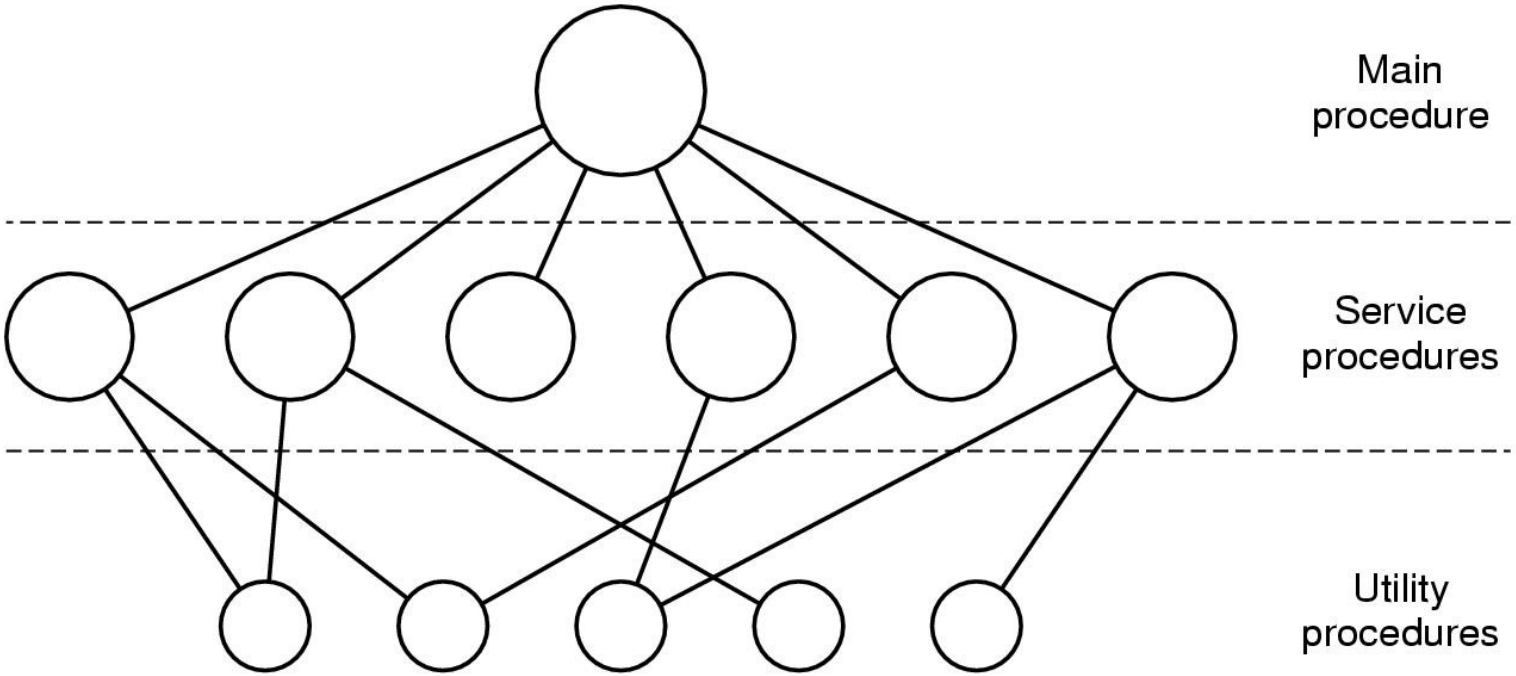
- Llamadas al sistema
  - Las llamadas al sistema se acceden generalmente a través de **funciones de biblioteca**
  - El programador generalmente accede a esta “interfaz”, en lugar de hacer la llamada al sistema real
  - Estas funciones de biblioteca, ponen realizan cierto trabajo previo, y luego ejecutan una instrucción especial tipo TRAP para pasar del modo usuario al modo kernel
  - Cuando se termina la ejecución de la llamada al sistema (lo cual puede requerir E/S), el SO hace un RETURN FROM TRAP para devolver el control al proceso de usuario

Unidad 1: Sistemas Operativos - Conceptos

- Llamadas al sistema



- Estructura de los sistemas operativos
  - Sistemas monolíticos

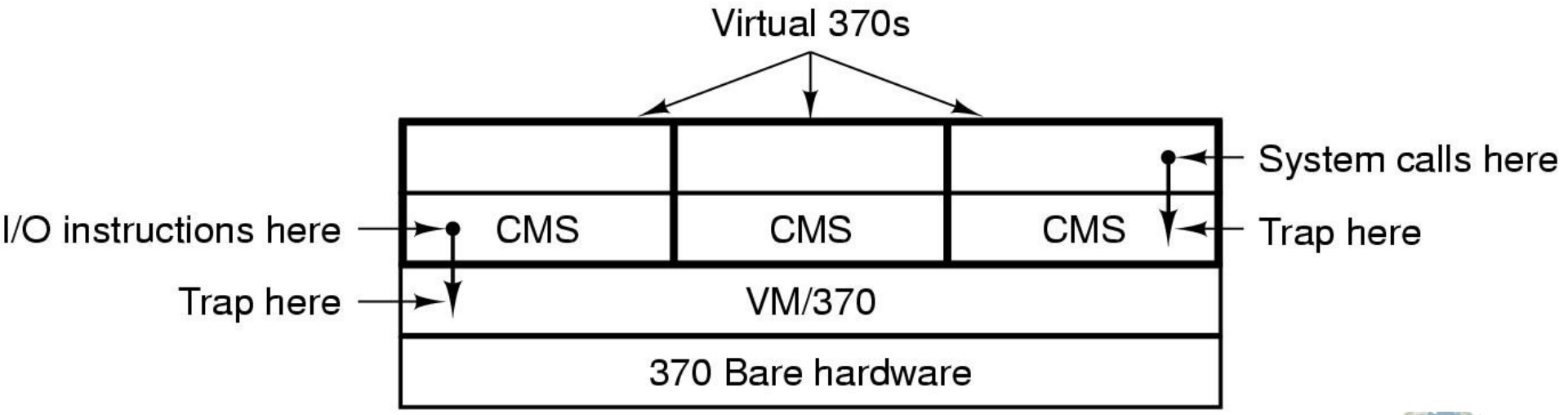


Unidad 1: Sistemas Operativos - Conceptos

- Estructura de los sistemas operativos
  - Sistemas en capas. Ej. THE (Dijkstra, 1968)

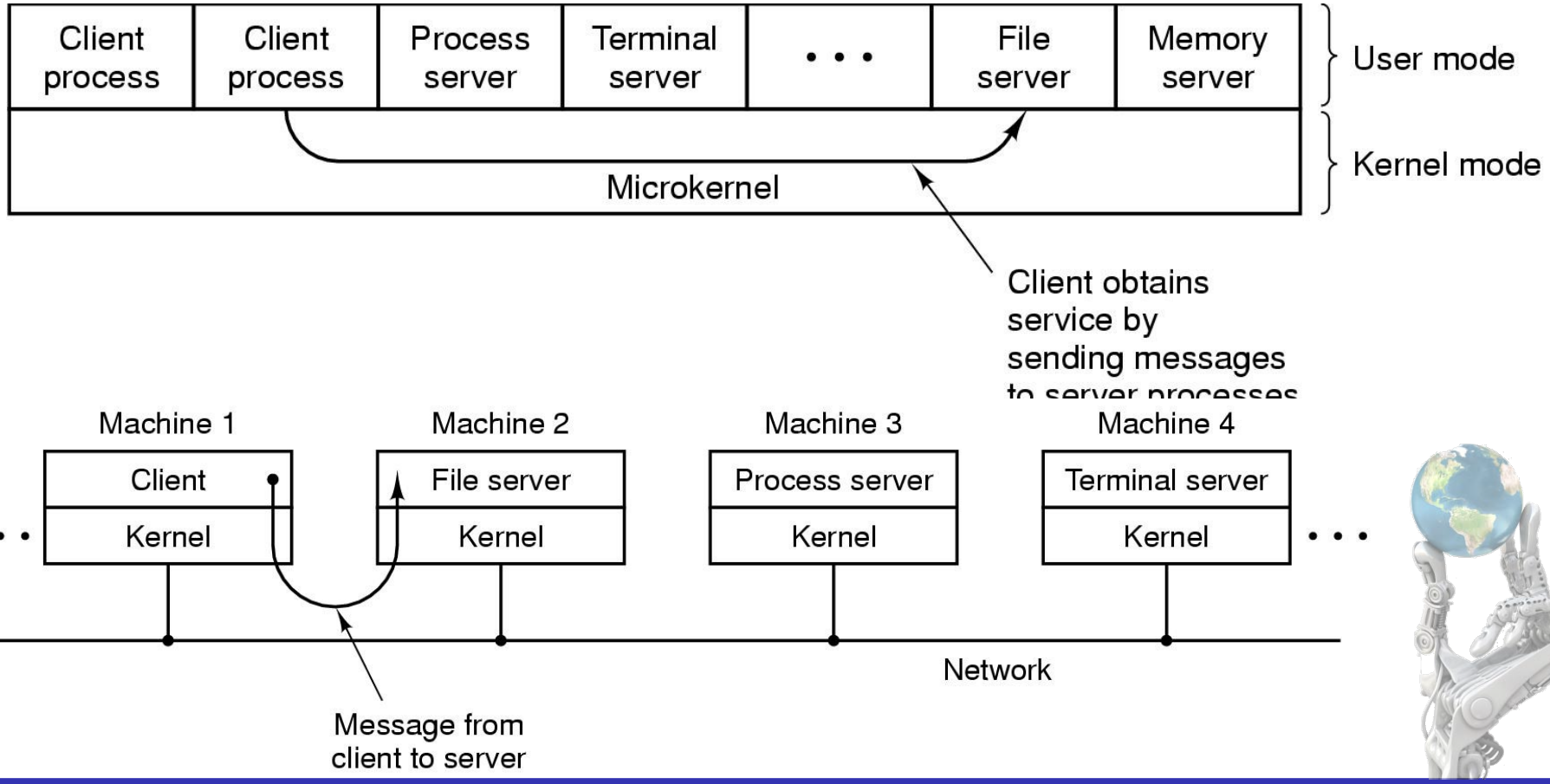
Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

- Estructura de los sistemas operativos
  - Máquinas virtuales



- Estructura de los sistemas operativos
  - Micro-kernel
    - La cantidad de errores (bugs) depende del tamaño del componente, su edad, etc.
    - Se estima que en sistemas de tamaño industrial, existen 10 errores cada 1000 lineas de codigo
    - En un sistema complejo, de 1.000.000 de lineas de codigo, habrían en el orden de 10.000 bugs
    - Los bugs a nivel de kernel son graves porque pueden tirar el sistema completo
    - En aplicaciones industriales se requiere alta disponibilidad y confiabilidad → un error en un módulo no debe comprometer el sistema completo. Ej: aviónica, sistemas militares, etc.
    - Una forma de reducir estos problemas es tener kernels lo más pequeños posible (micro-kernels), y dejar lo más posible en espacio de usuario → mover algunas funciones del sistema operativo al espacio de usuario.

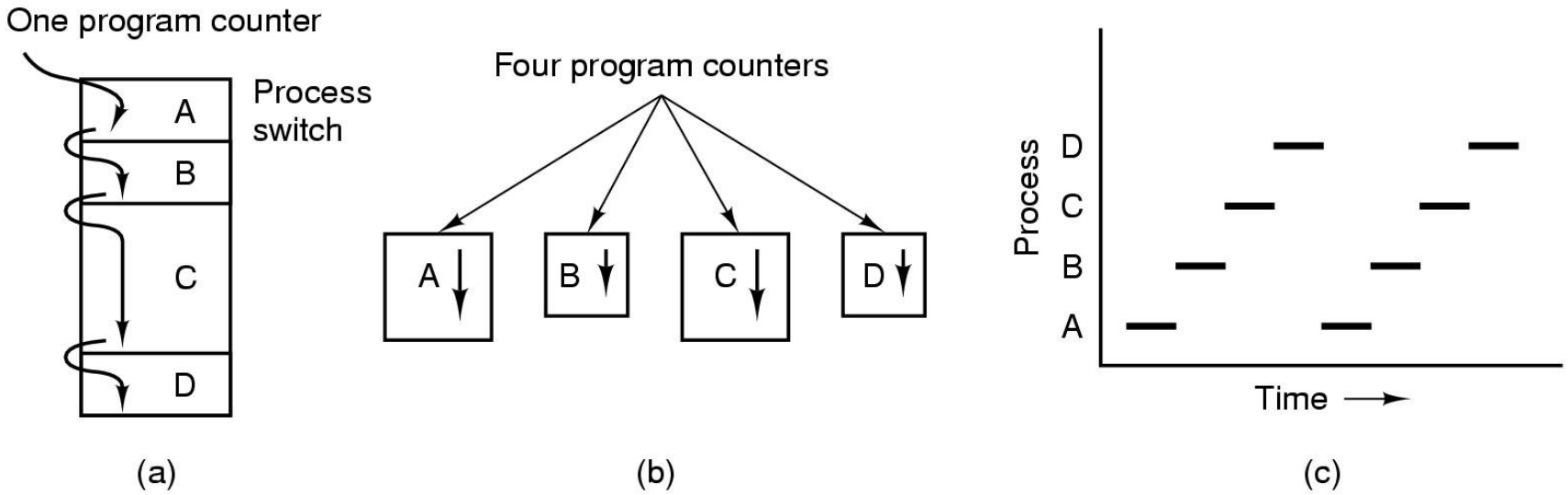
- Estructura de los sistemas operativos
  - Modelo Cliente/Servidor



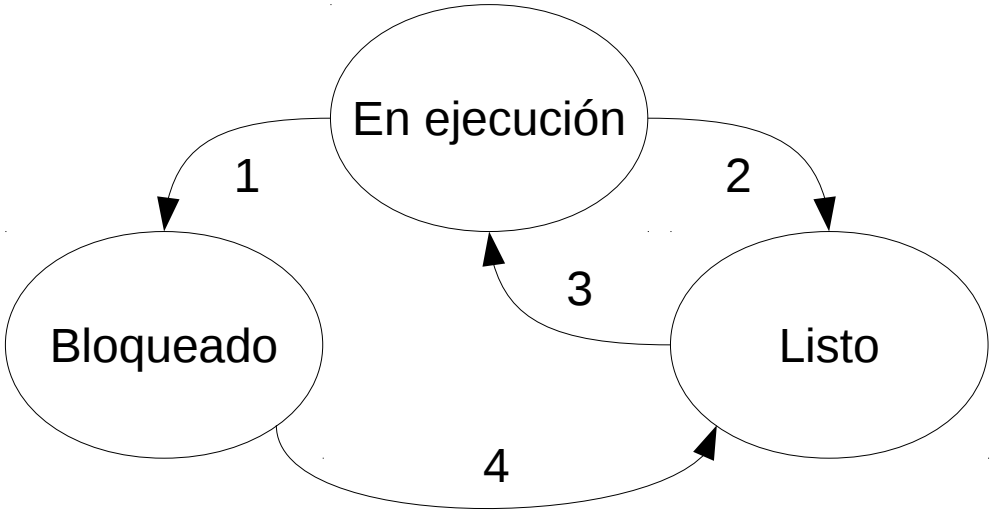
Procesos

- Multiprogramación
  - Una CPU solo puede ejecutar un proceso a la vez
  - Para lograr la aparienca de paralelismo (es decir, el ***pseudo-paralelismo***), el sistema operativo alterna rápidamente entre los procesos que se están ejecutando en un momento dado
  - A esto se lo llama ***multiprogramación***
  - La ventaja de la multiprogramación es que se aprovecha más la CPU cuando algún proceso está esperando por algún recurso
    - Ej:
      - Si un proceso tiene que leer información del disco o esperar algo que viene por la red, ese proceso estará ***ocioso*** hasta recibir sus datos
      - Ese tiempo ocioso lo puede aprovechar otro proceso

- Multiprogramación



- Estados de un proceso
  - En ejecución: el proceso está ocupando la CPU actualmente
  - Bloqueado: el proceso está esperando por algún recurso (lectura de disco, red, etc.); no puede continuar por ahora
  - Listo: el proceso no está ocupando la CPU, pero está listo para hacerlo (no está esperando nada)



1. El proceso se bloquea
2. El S.O. elige otro proceso
3. El S.O. elige este proceso
4. Los datos están disponibles

- Estados de un proceso
  - La decisión de bloquearse o no, la toma cada proceso
  - Si un proceso se bloquea, el S.O. automáticamente lo deja de ejecutar y guarda su estado hasta que esté listo nuevamente
  - Un componente del SO se encarga del bloqueo, suspensión, reanudación, etc. de los procesos: **el planificador**
  - Los propios componentes del SO generalmente se estructuran como procesos, y el planificador les da el uso de la CPU, tal como lo hace con los procesos de usuario
  - Cuando hay múltiples procesos listos, el planificador va rotando entre ellos de acuerdo a alguna política determinada (ej: una lista cíclica: round robin). Esta política puede además incluir prioridades (un proceso de kernel tendrá mayor prioridad que un proceso de usuario).

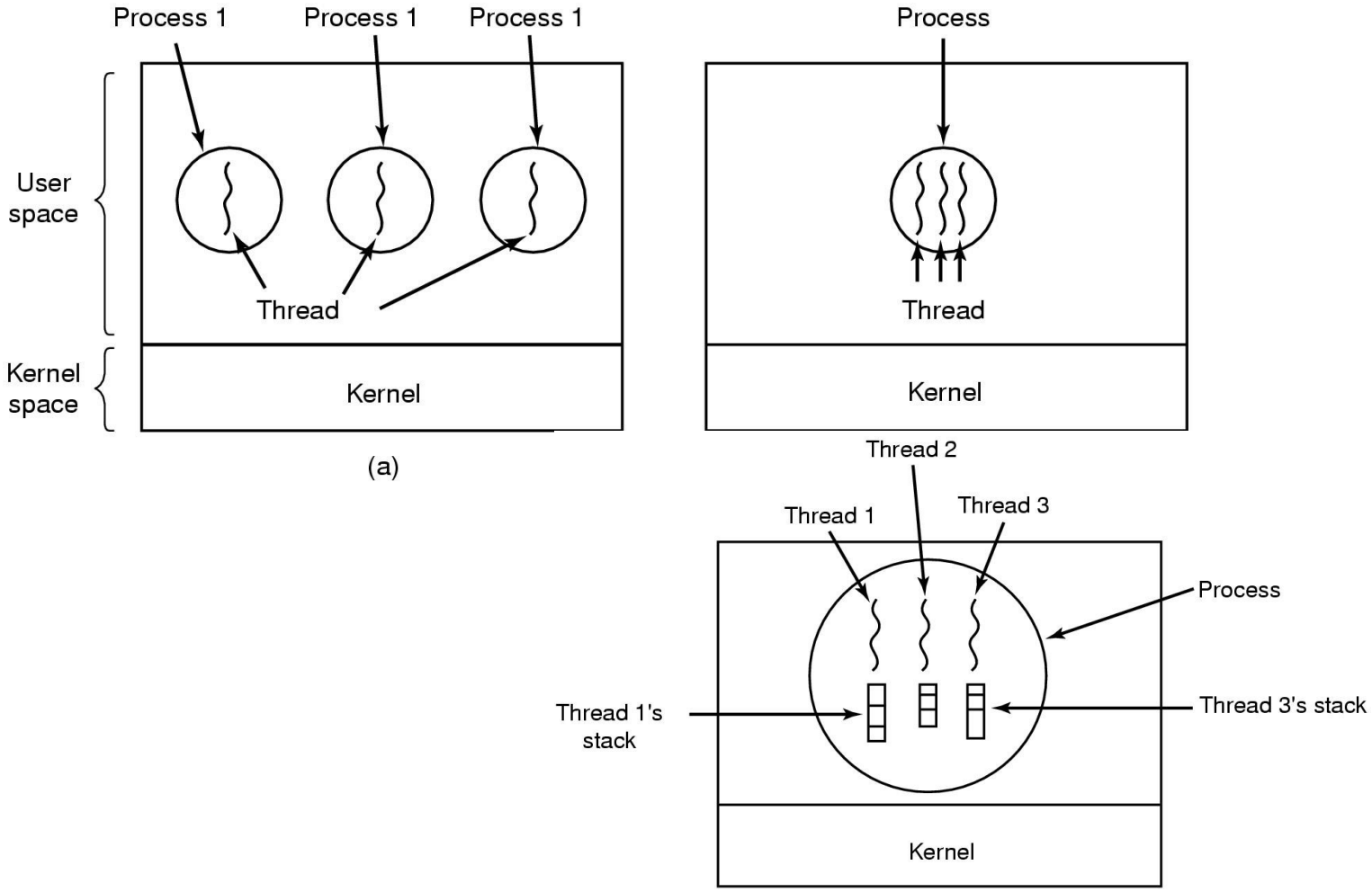
Unidad 1: Sistemas Operativos - Procesos

- Hilos (threads): También llamados ***procesos ligeros***
  - Los threads son *hilos de ejecución* paralelos
  - Cada proceso tiene un espacio de direcciones independiente del de los otros procesos
  - En cambio, todos los hilos de un mismo proceso
    - Comparten las mismas instrucciones
    - Comparten el mismo espacio de memoria (variables)
    - Pero cada uno tiene un valor independiente para el **PC**, **registros**, **pila** y **estado**, con lo cual la ejecución de cada uno puede variar

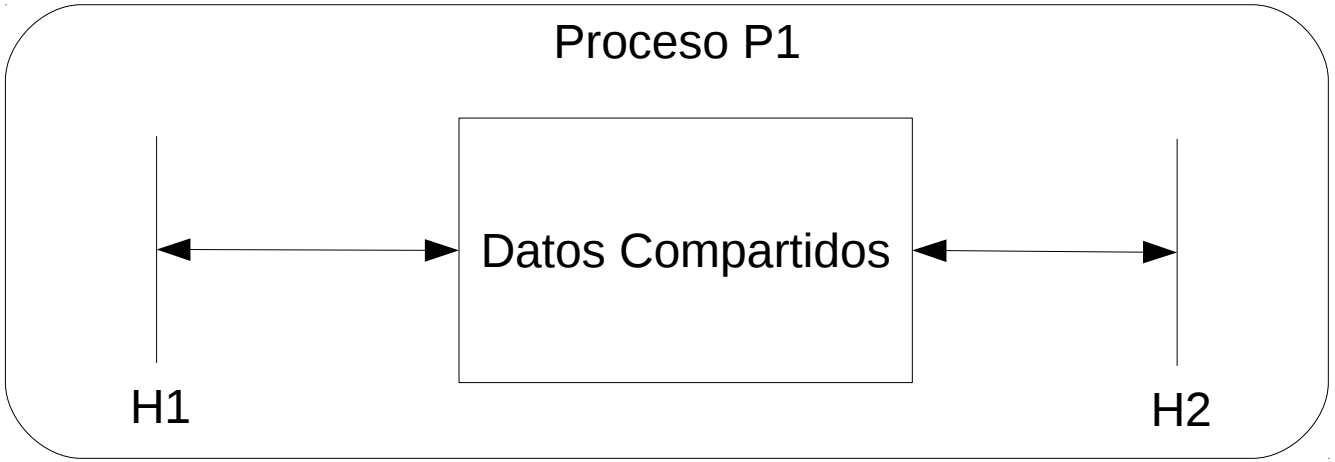
- Los threads múltiples comparten algunas de las características de los procesos múltiples:
  - Pueden estar en los mismos estados básicos (en ejecución, bloqueado, listo)
  - Permiten aprovechar mejor la CPU cuando un hilo está bloqueado esperando por datos del disco o la red, u otro hilo o proceso

Unidad 1: Sistemas Operativos - Procesos

- Threads: relación con los procesos

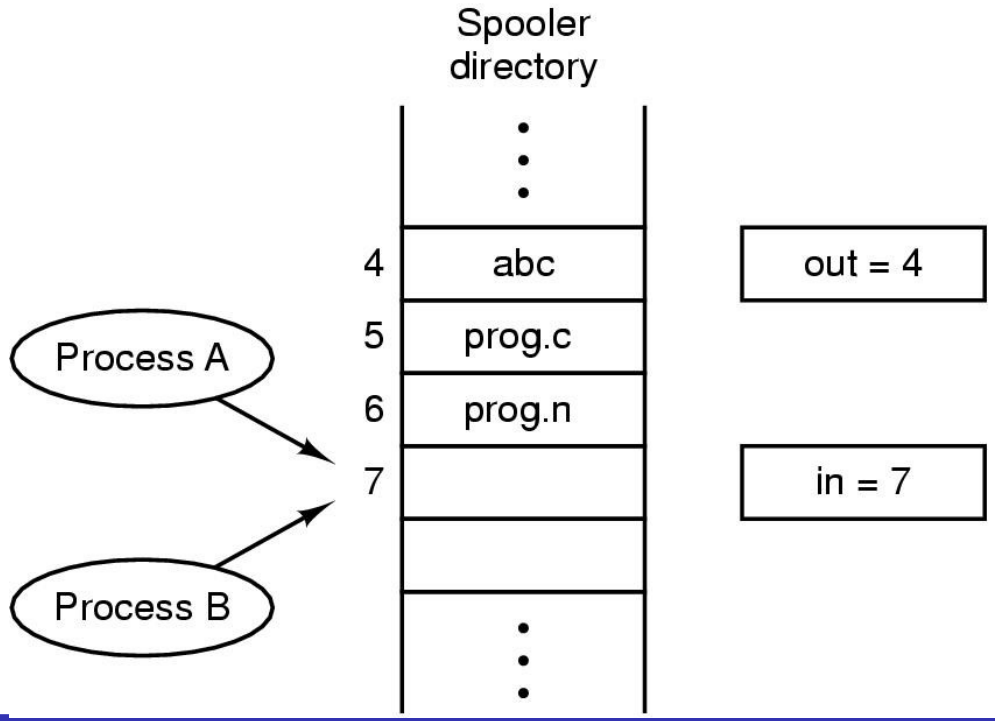


- El hecho de que los threads comparten el espacio de memoria hace más simple el intercambio de datos entre los hilos



- En cambio, la comunicación entre procesos independientes generalmente requiere el uso de algún mecanismo explícito. Ej: pipes
- En ambos casos, la sincronización es muy importante para evitar problemas de concurrencia

- Condiciones de competencia
  - Los hilos/procesos pueden tener problemas cuando acceden a recursos compartidos
  - Estas situaciones se llaman ***condiciones de competencia***



Unidad 1: Sistemas Operativos - Procesos

- Sincronización
  - Existen partes del código que no presentan riesgo ante la concurrencia, mientras que otras áreas del código pueden potencialmente generar problemas
  - Las áreas del código problemáticas son las que leen/modifican estructuras de datos compartidas o acceden a recursos compartidos: estas son las llamadas ***secciones críticas***
  - Nunca 2 procesos o threads deben estar en la misma sección crítica al mismo tiempo, puesto que los resultados son impredecibles
- Un mecanismo de sincronización típico es el uso de ***semáforos***
- Los semáforos son variables especiales que pueden ser modificados con funciones especiales mediante **operaciones atómicas**
- La idea es que antes de entrar a una sección crítica, el thread o proceso debe tratar de “tomar” el semáforo; si el semáforo está libre, el thread/proceso puede entrar en su sección crítica; si no, se bloquea esperando que el proceso que tiene el semáforo lo libere

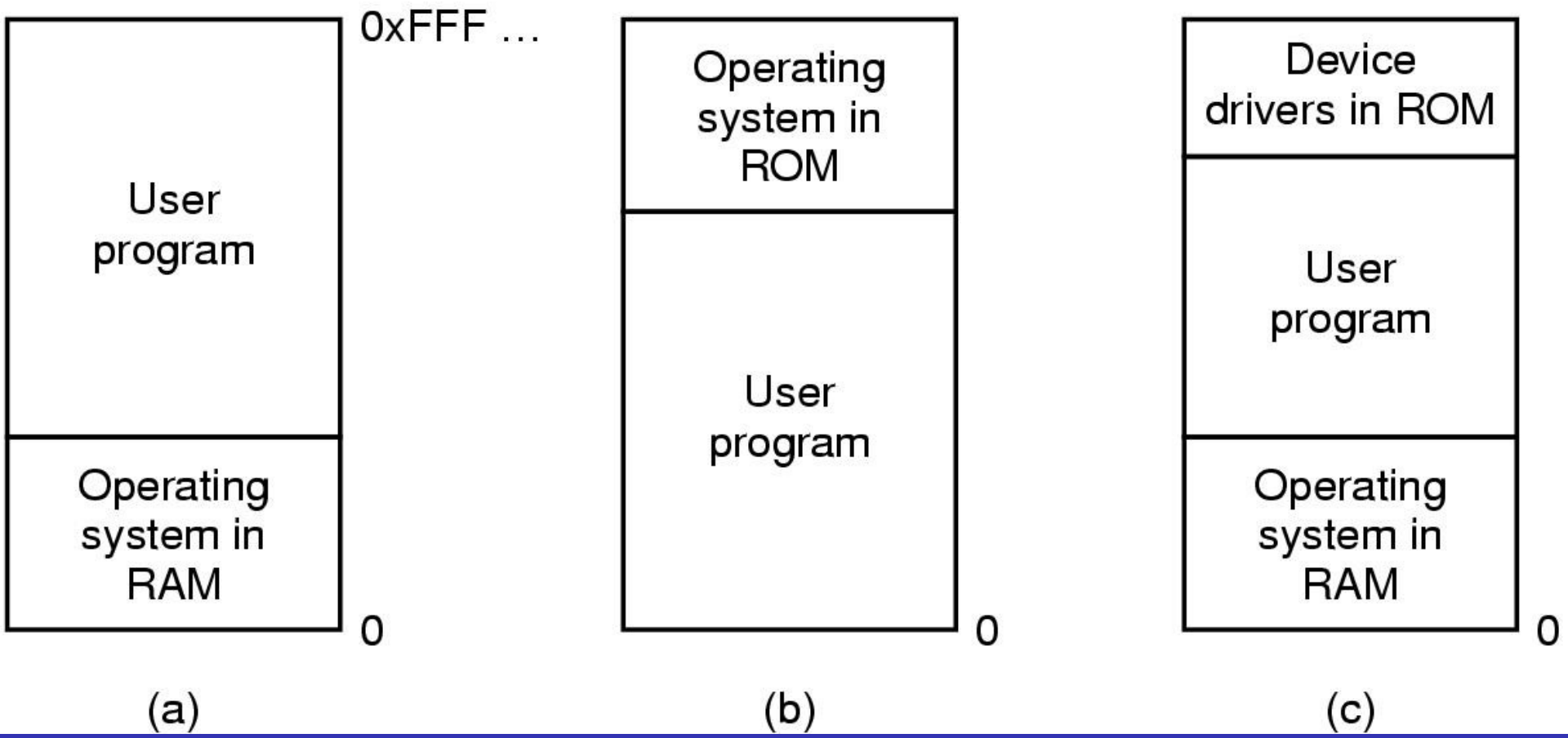
Memoria

Unidad 1: Sistemas Operativos - Memoria

- El SO tiene un componente que gestiona la memoria: ***administrador de la memoria***
  - Lleva registro de las partes de la memoria utilizada y libre
  - Asigna espacio a procesos cuando se lo requiere, y libera el espacio cuando se termina
  - Maneja la ***memoria virtual*** (intecambio)

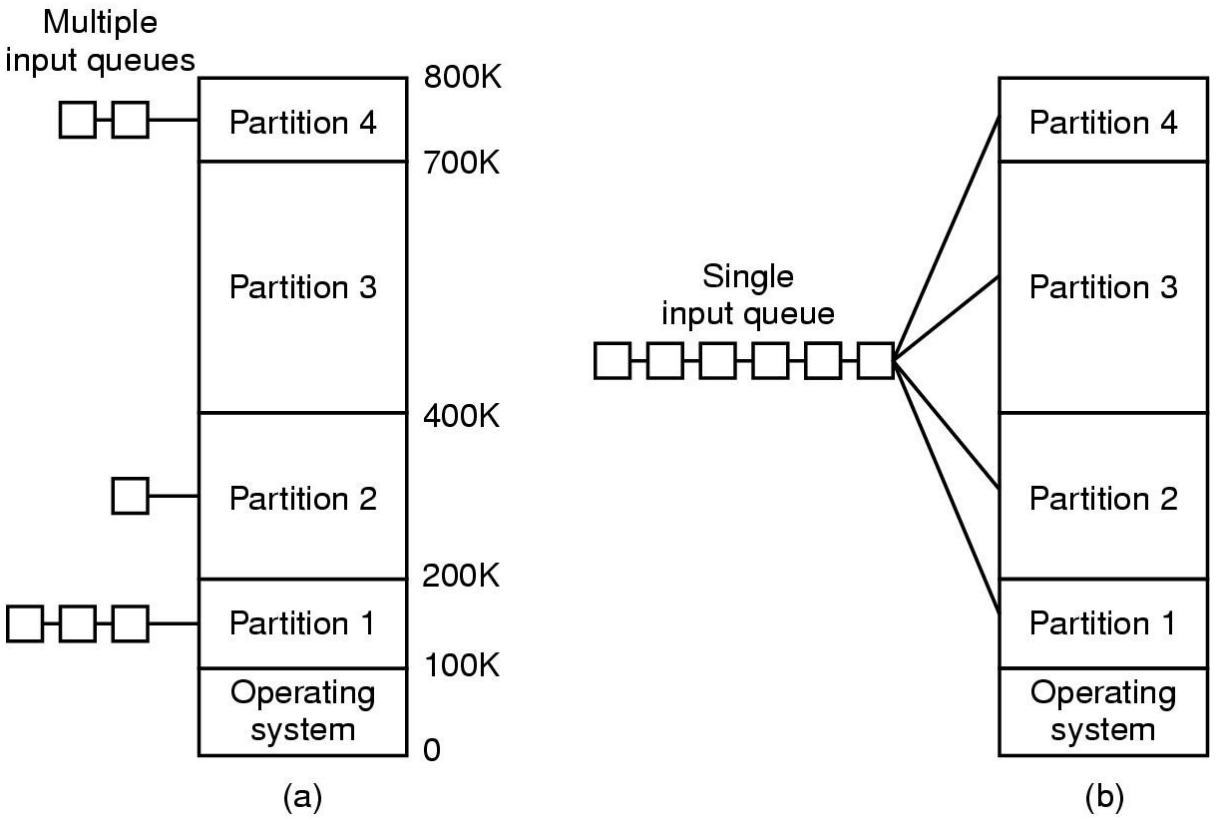
Unidad 1: Sistemas Operativos - Memoria

- Monoprogramación sin intercambio ni paginación



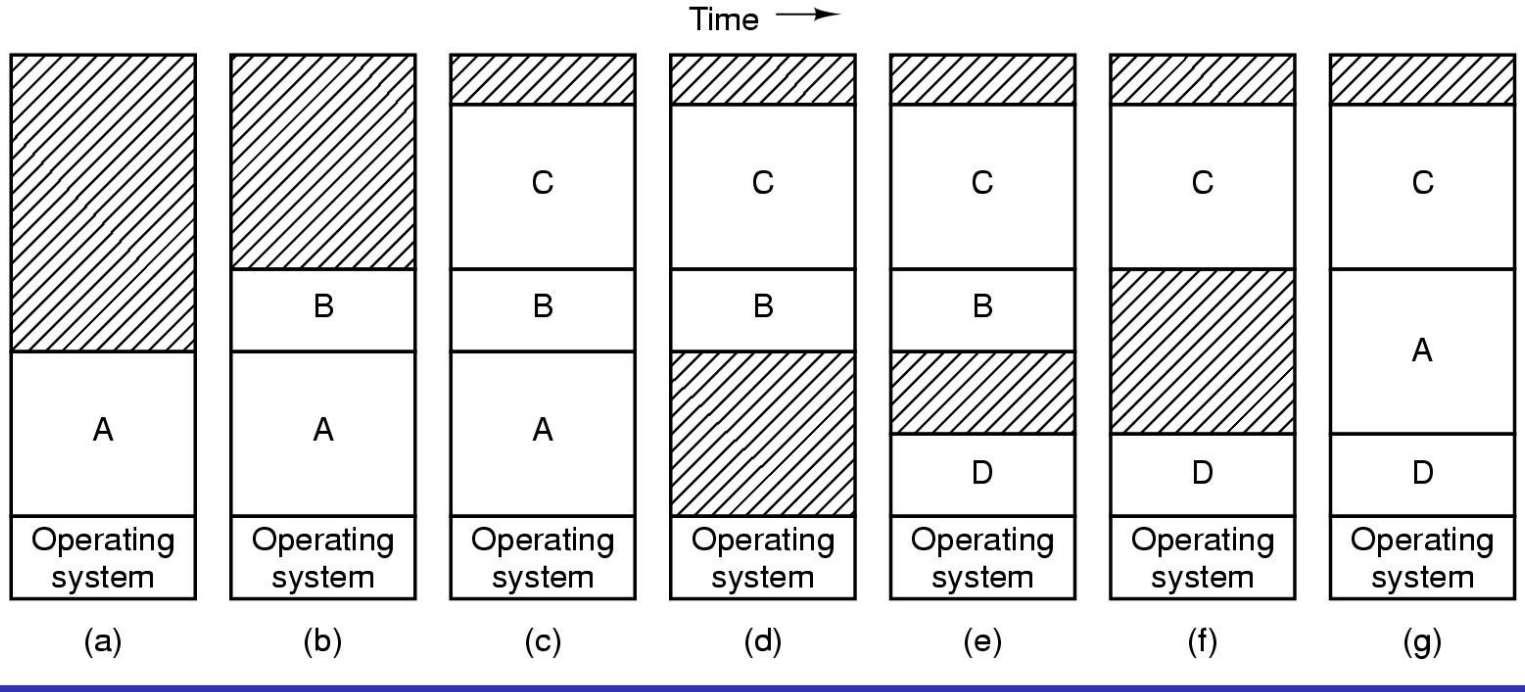
Unidad 1: Sistemas Operativos - Memoria

- Multiprogramación con particiones fijas
  - Los procesos se encolan en la partición de tamaño fijo más pequeña que satisface el requerimiento



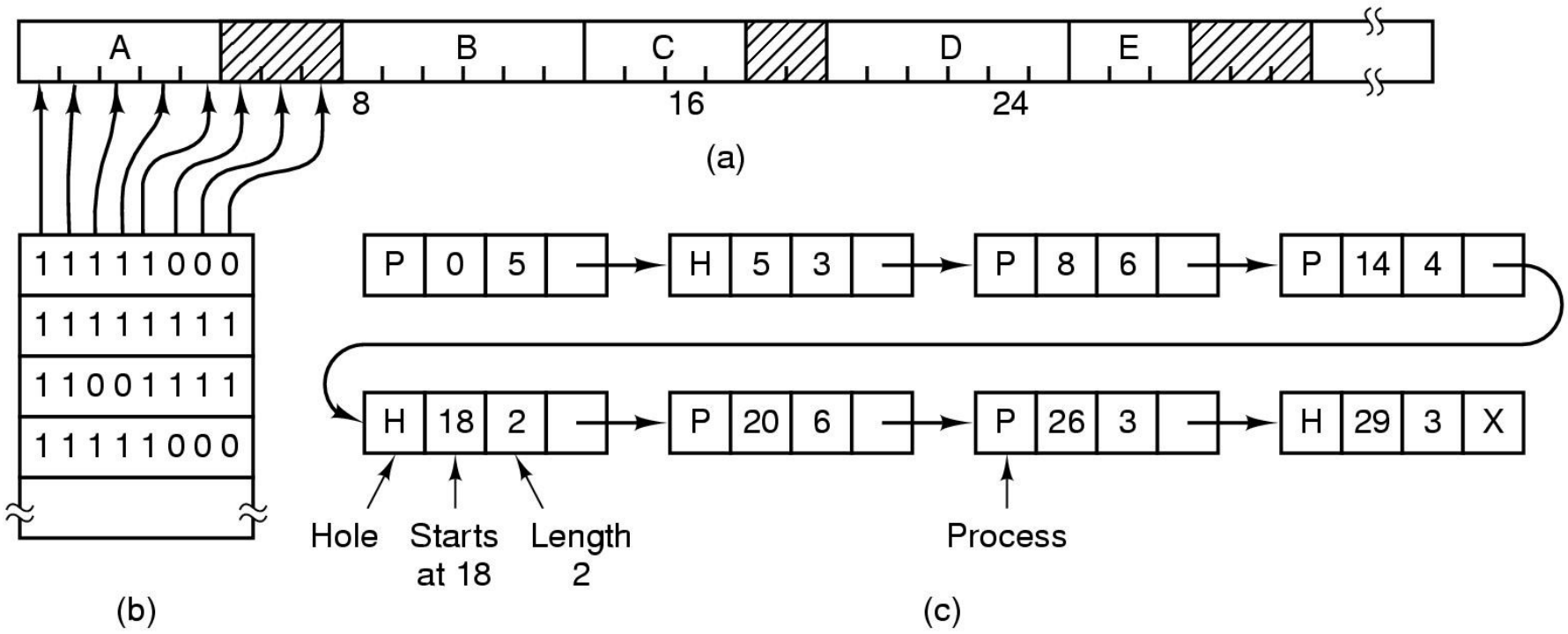
Unidad 1: Sistemas Operativos - Memoria

- Intercambio: multiprogramación con particiones variables
  - La cantidad, tamaño y ubicación de las particiones es dinámica: se establece en el momento necesario
    - Potencial problema: fragmentación de la memoria
    - Solución: compactación de la memoria (solo de ser necesario, porque es ineficiente)

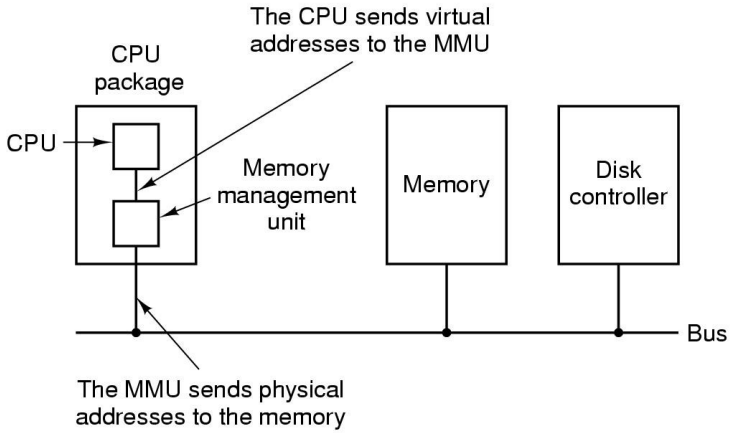


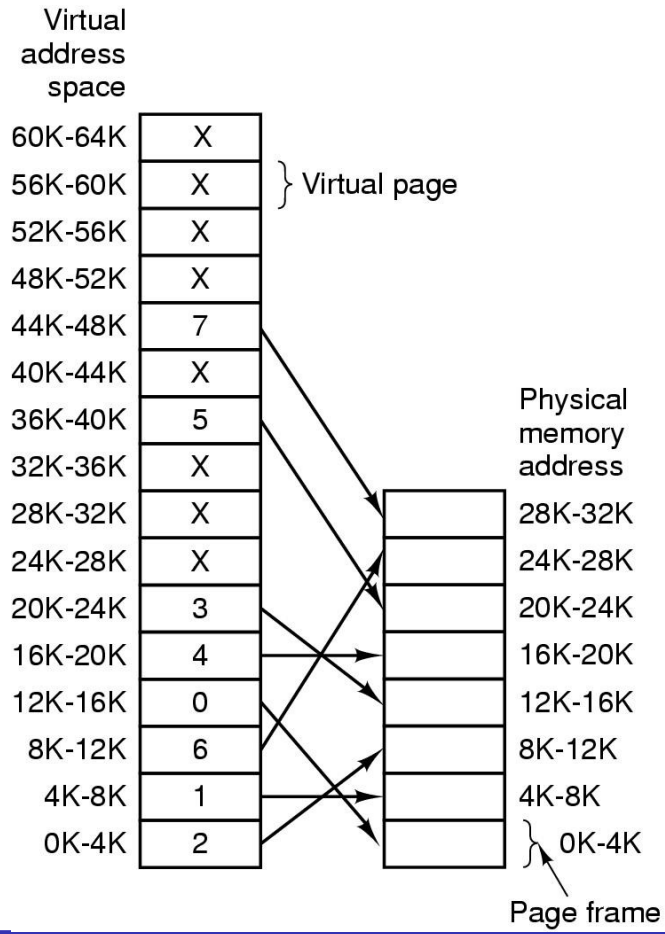
Unidad 1: Sistemas Operativos - Memoria

- Administración de la memoria: algunos mecanismos
  - Mapas de bits
  - Listas enlazadas



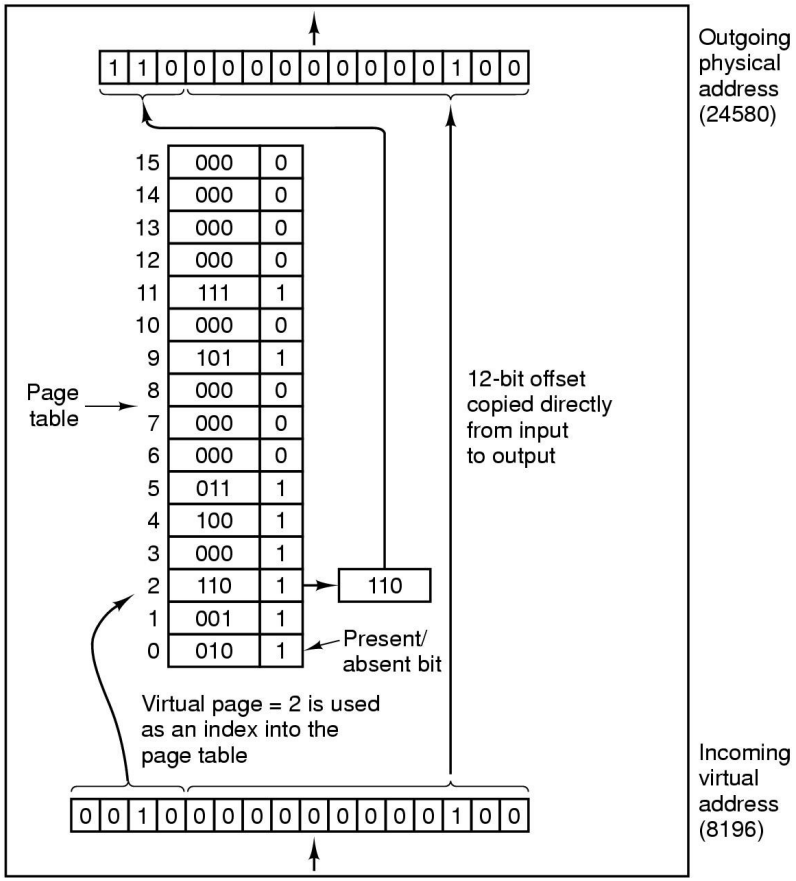
- Memoria virtual: Paginación
  - La implementan el hardware y el SO
  - Permite direccionar un espacio de memoria mayor que la memoria física disponible
  - Componentes:
    - Generalmente un chip convierte direcciones virtuales en físicas (Memory Management Unit, **MMU**)
    - Cuando una página no está en memoria, la MMU notifica al SO, quien ejecuta el ***intercambio*** **(*swapping*)**





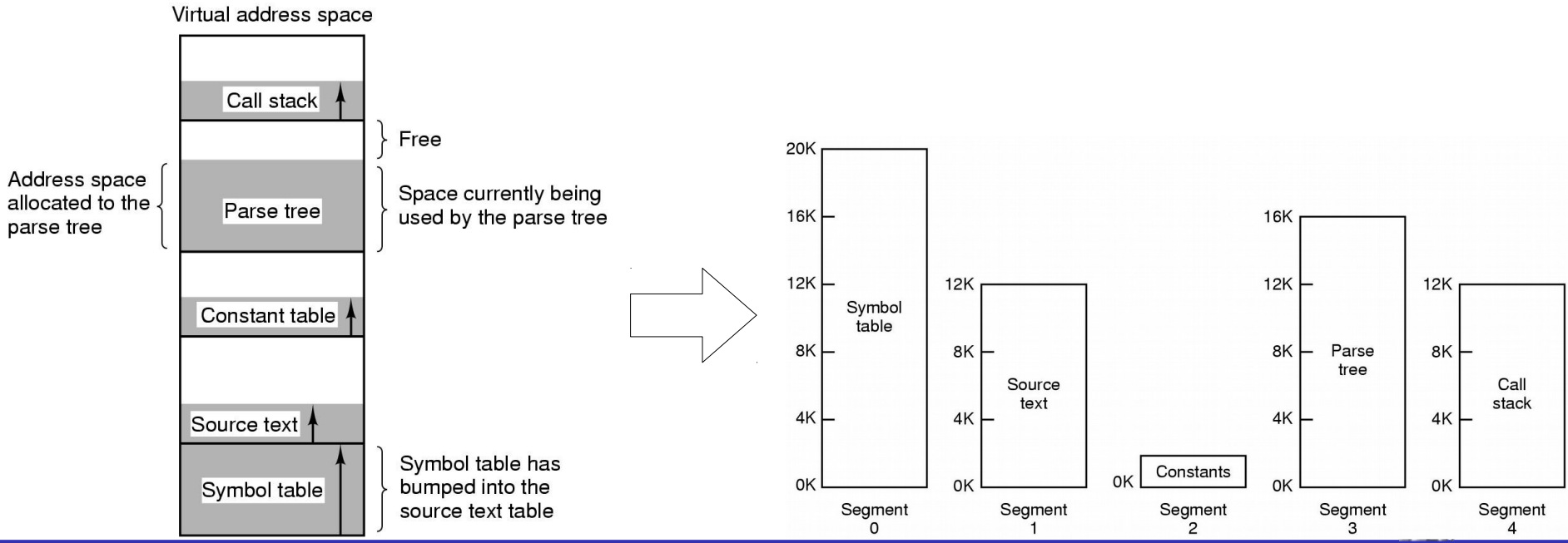
Unidad 1: Sistemas Operativos - Memoria

- Memoria virtual: Paginación
  - Algoritmo de mapeo de direcciones virtuales a direcciones físicas



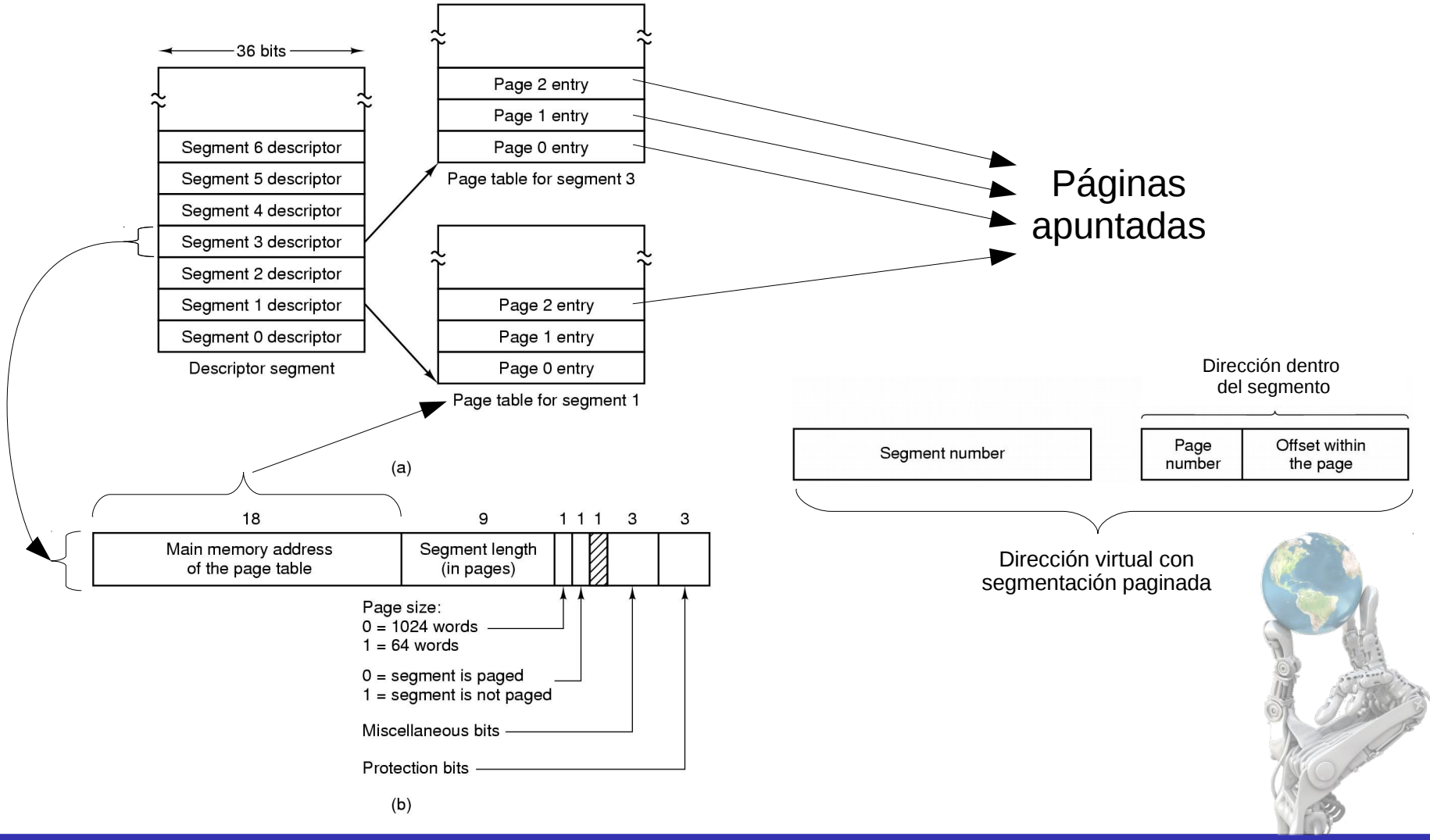
Unidad 1: Sistemas Operativos - Memoria

- Segmentación
  - Permite tener espacios “virtuales” de direccionamiento independientes
  - Utilidad
    - Cuando se necesitan porciones de memoria que crezcan de manera independiente y variable
    - También permite aplicar protección mediante permisos, para que un proceso no pueda leer/escribir segmentos que no son suyos



Unidad 1: Sistemas Operativos - Memoria

- Segmentación: Ejemplo de implementación de segmentación paginada



Sistema de archivos

Unidad 1: Sistemas Operativos - Archivos

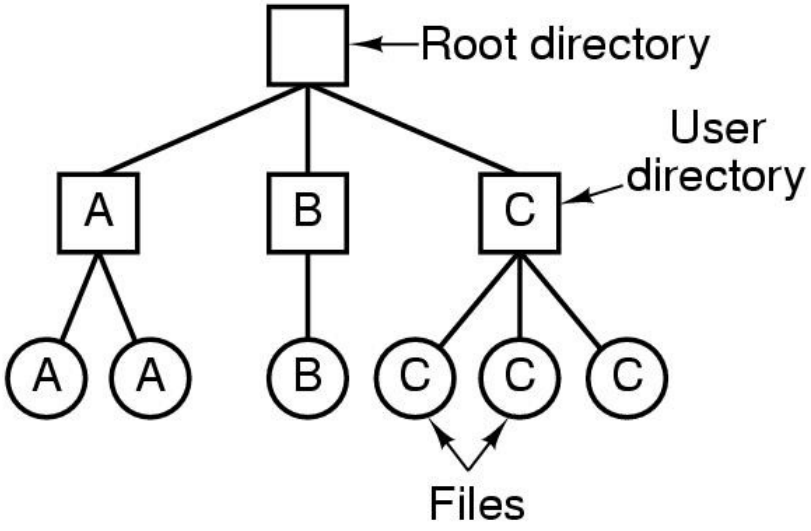
- Los archivos permiten leer/escribir información y mantenerla de manera persistente
- Para el SO, un archivo es un conjunto de bytes:
  - El SO no prescribe ni exige una estructura interna específica para los archivos
  - La estructura de los archivos queda en manos de los programas de usuario (ej: archivos de texto, archivos binarios secuenciales, árboles, etc)
- Algunos sistemas de archivos distinguen mayúsculas de minúsculas (***case sensitive***)
- Los archivos se nombran mediante un nombre y una extensión (en algunos sistemas la extensión es obligatoria; en otros es opcional)

Unidad 1: Sistemas Operativos - Archivos

- Existen básicamente 4 tipos de archivos:
  - Archivos regulares
  - Directorios
  - Archivos especiales de bloques
  - Archivos especiales de carácter
- Modos de acceso: dependen del tipo de archivo y el dispositivo asociado
  - Secuencial:
    - Flujo de bytes debe ser leido en orden
    - Ej: archivo especial para un dispositivo de red sin buffer
  - Aleatorio:
    - Puede leerse cualquier bloque del archivo sin leer los bloques previos
    - Ej: la mayoría de los archivos regulares y dispositivos (discos rígidos)

Unidad 1: Sistemas Operativos - Archivos

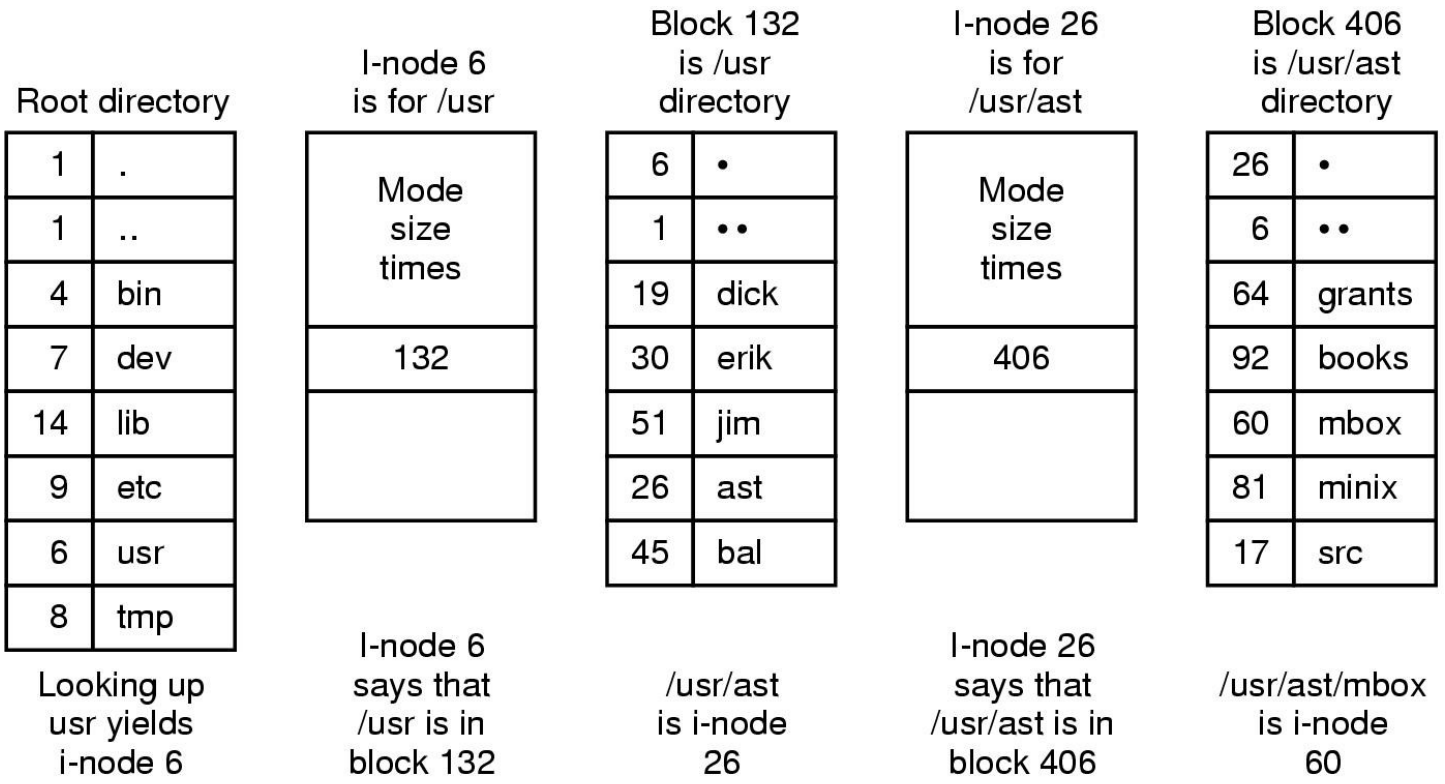
- Directorios
  - En muchos sistemas, los directorios también son archivos
  - En lugar de contener bytes de datos de usuario, contienen listas de archivos agrupados bajo ese directorio
  - En la mayoría de los casos, los directorios se organizan en un árbol de directorios



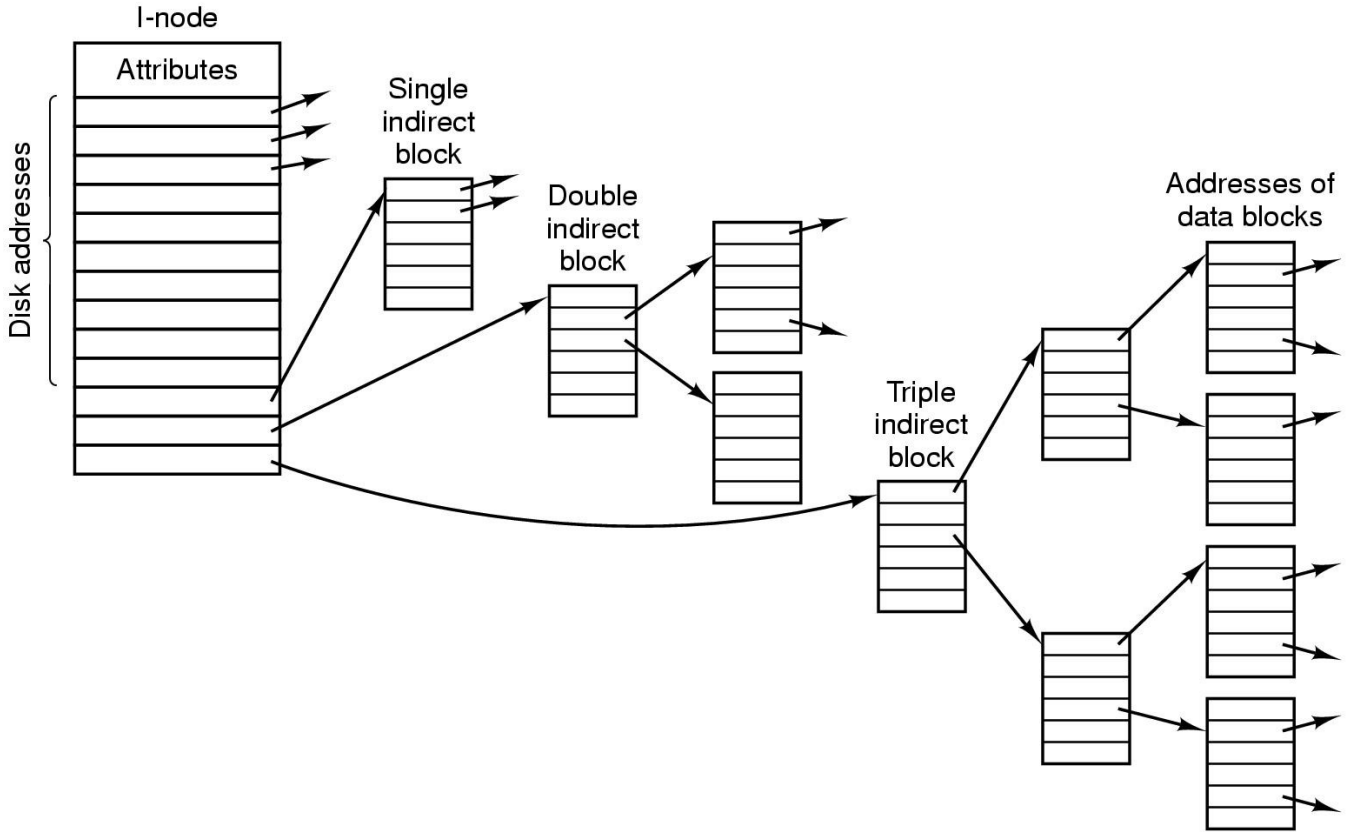
Unidad 1: Sistemas Operativos - Archivos

- Implantación de directorios
  - El directorio tiene por finalidad ubicar el contenido de los archivos que se encuentran almacenados en el disco
  - En la mayoría de los sistemas, cada entrada de directorio contiene una lista de los archivos allí almacenados, y un apuntador a la posición de los mismos en el disco
  - Ej: Directorio en sistemas UNIX (en general, sistemas basados en nodo-i)
  - Los nodo-i tienen posiciones fijas en el disco. El nodo-i del directorio raíz es fijo en el sistema de archivos

- Implantación de directorios
  - Ej: Directorio en sistemas UNIX (en general, sistemas basados en nodo-i)



- Implantación de archivos
  - Una vez localizado el nodo-i del archivo, se debe poder acceder a todos los trozos del mismo (pueden no estar almacenados en forma contígua)



Entrada/Salida

Unidad 1: Sistemas Operativos – E/S

- El SO debe proveer una interfaz simplificada a los programas de usuario para acceder a los dispositivos de E/S
- Aspectos relevantes
  - Debe proveer nombres uniformes, no importa el tipo de dispositivo de que se trate
  - Manejo de errores lo más cerca del hardware posible
  - Simplificación del modelo de transferencia
    - La mayoría de la E/S se basa en interrupciones (asíncrona)
    - Pero el modelo de programación es más simple si se logra una transferencia síncrona
  - Gestión de recursos que pueden compartirse ***concurrentemente*** (ej: el disco) y recursos que son de uso exclusivo (ej: la impresora)

Unidad 1: Sistemas Operativos – E/S

- Para cubrir esos aspectos relevantes, el software de E/S se estructura en 4 capas:
  - Manejadores de interrupciones
  - Manejadores de dispositivos (***device drivers***)
  - Software de E/S independiente del dispositivo
  - Software de E/S en espacio de usuario