Tema 1

Fundamentos de los SO

3º ETSI

Informática

Contenido

Tema 1. Fundamentos de los SO. 4h

1.1 Concepto de sistema operativo

El sistema operativo como una máquina virtual

1.2 Soporte hardware para SO

Interrupciones

Reloj

Jerarquía de memoria

1.3 Evolución histórica de los sistemas operativos

Generaciones de SO y primeros conceptos

1.4 Características y funciones del sistema operativo

Implementación del SO

1.5 Estructuras de SOs

1.1 Concepto de SO

1.1 Concepto de SO

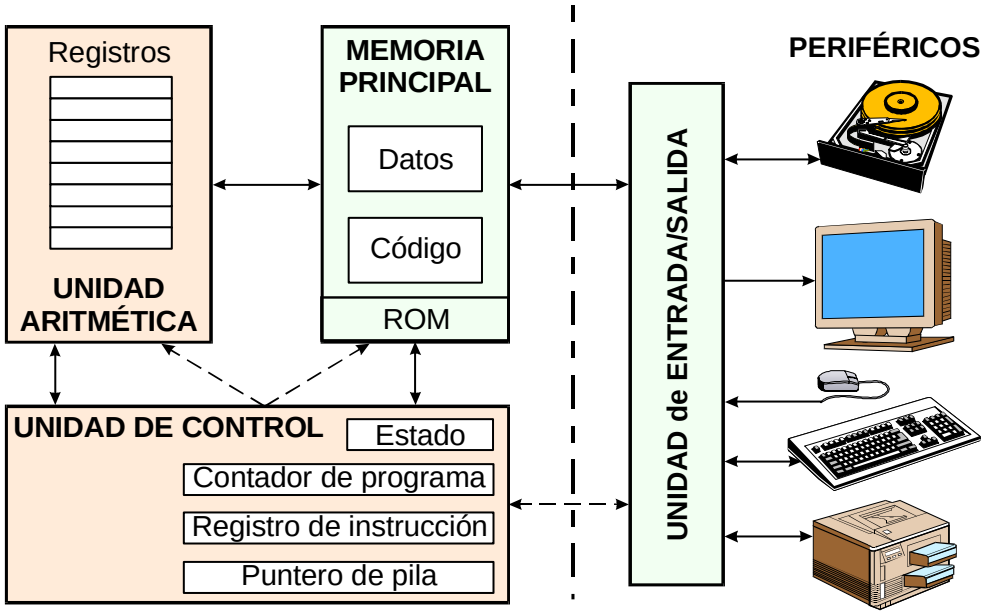
- Definiciones:

▪ Conjunto de programas, implementados como software o firmware, que permiten utilizar el HW del computador a todos los usuarios, alcanzando buenas prestaciones.

• Un programa que actúa como intermediario entre el usuario de un computador y el hardware del mismo

1.1 Concepto de SO

El SO viste a “la máquina desnuda”



Fuente: SO Una visión Aplicada, Fig 1.2

La “máquina desnuda” es el computador sin SO.

Sólo es capaz de repetir a alta velocidad la secuencia de:

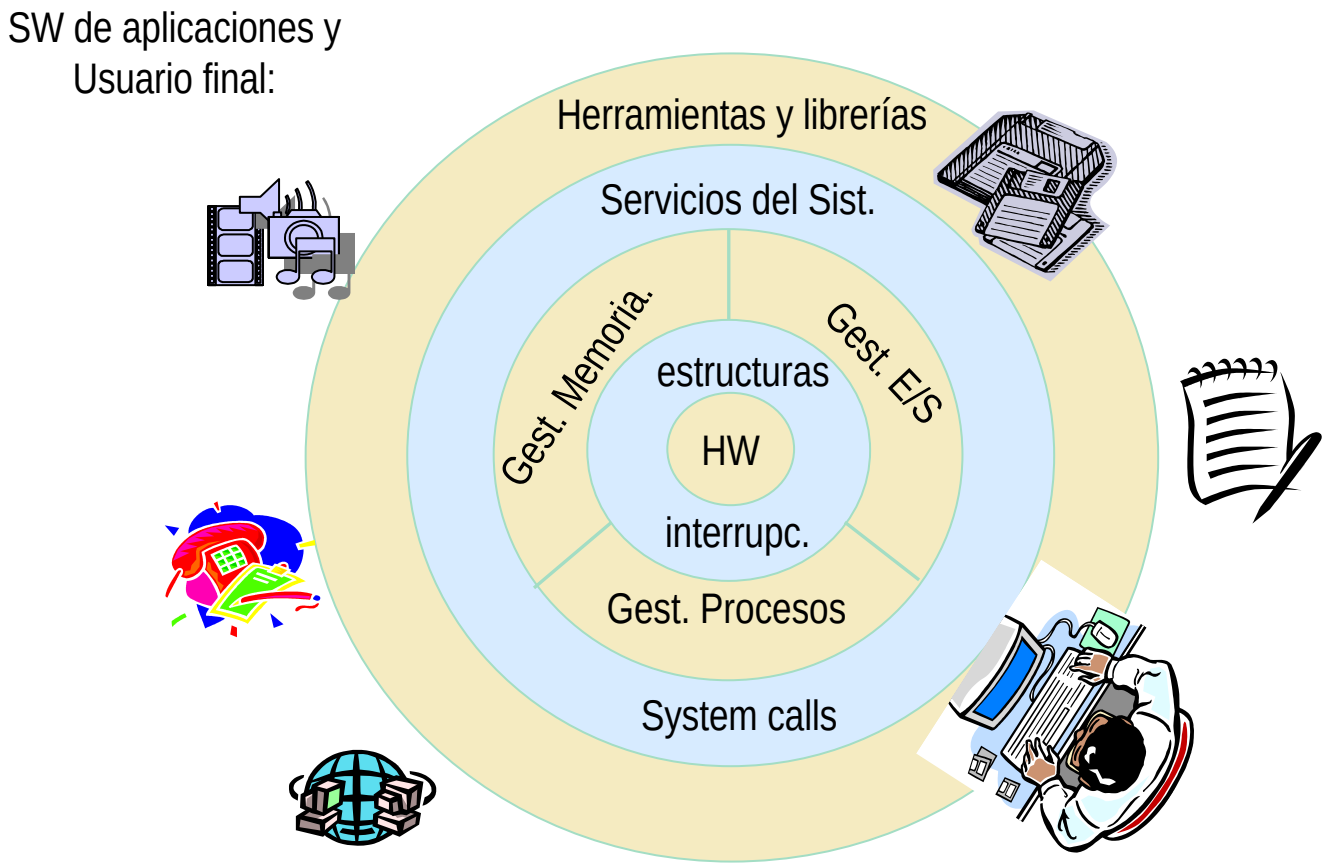
1) Lectura de Inst, 2) Incremento PC, 3) Ejecución Inst.

1.1 Concepto de SO

sistema informático

- El SO ocupa un lugar dentro del "Sistema Informático"

  Conjunto de HW y SW para el tratamiento automático de la información



1.1 Concepto de SO

niveles del sistema informático

- Usuarios y aplicaciones
- SW de base
  - Herramientas y librerías
    - Editores
    - Compiladores
    - Shell: interprete de comandos
  - SO
    - System calls / API
    - Gestión
      - CPU
      - MEM
      - E/S
    - Interfaz con el HW
- HW

1.1 Concepto de SO

objetivos del SO

- Objetivos del SO:
  - Proporcionar al usuario una máquina “amigable”
  - Gestionar eficientemente el HW del computador
- Visiones del SO:

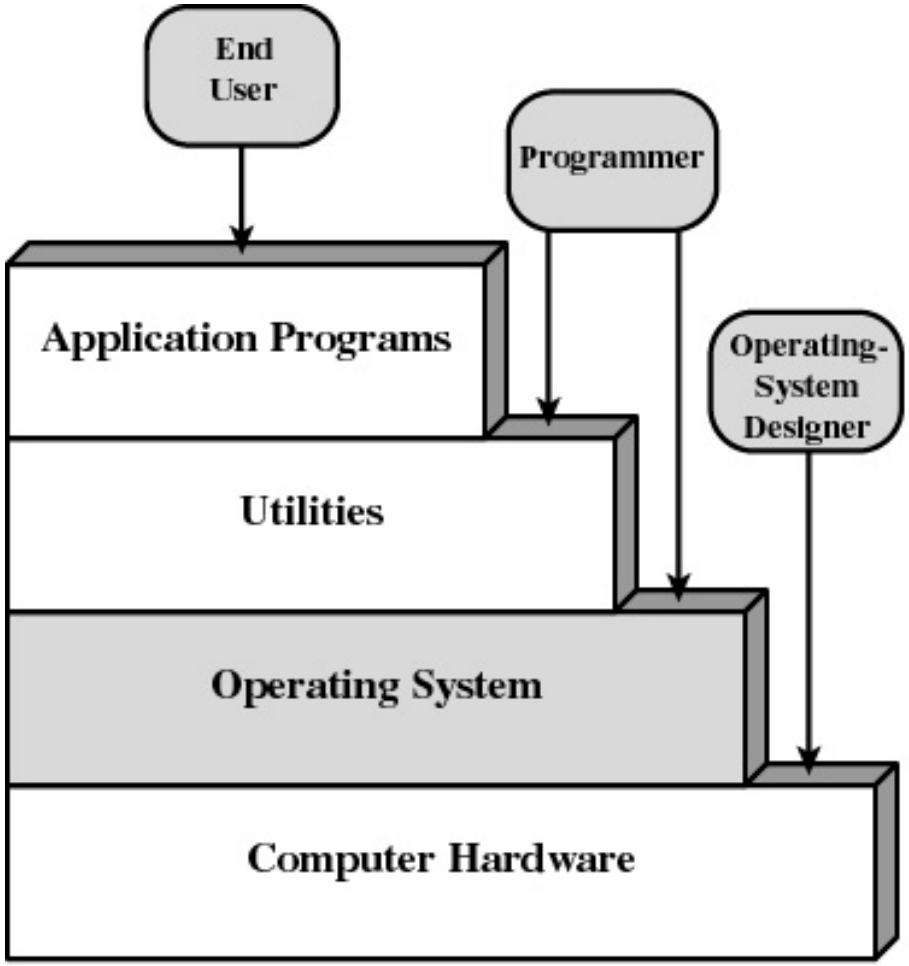
  - Descendente: Máquina extendida
    - Miras la máquina desde arriba y ves una máquina virtual fácil de usar
    - El SO proporciona las herramientas para el uso cómodo del HW
    - Sin el SO tendríamos que programar la máquina desnuda

  - Ascendente: Gestión del HW
    - Miras la máquina desde el punto de vista del HW (hacia arriba)
    - Ves un SW que gestiona y administra ese HW (CPU, MEM y E/S)
    - EL SO es el “gobierno” del HW
    - El SO debe administrar los recurso sin consumirlos

1.1 Concepto de SO

El SO como Interfaz Usuario/Computador

Niveles del Computador



Layers and Views of a Computer System

1.1 Concepto de SO

funciones del SO

- Funciones del SO
  - Gestión de los recursos de la comput.
    - Gestión del Procesador: gestión de procesos
    - Gestión de memoria
    - Gestión de E/S
  - Ejecución de servicios para programas
    - Ofrece un conjunto de servicios: system calls
    - Proporciona una máquina extendida
  - Interfaz con el usuario
    - Proporciona un interprete de mandatos
      - Espera una orden (texto o evento de ratón)
      - Analiza la orden y si es correcta la ejecuta
      - Concluida la orden vuelve a la espera

1.2 Soporte HW

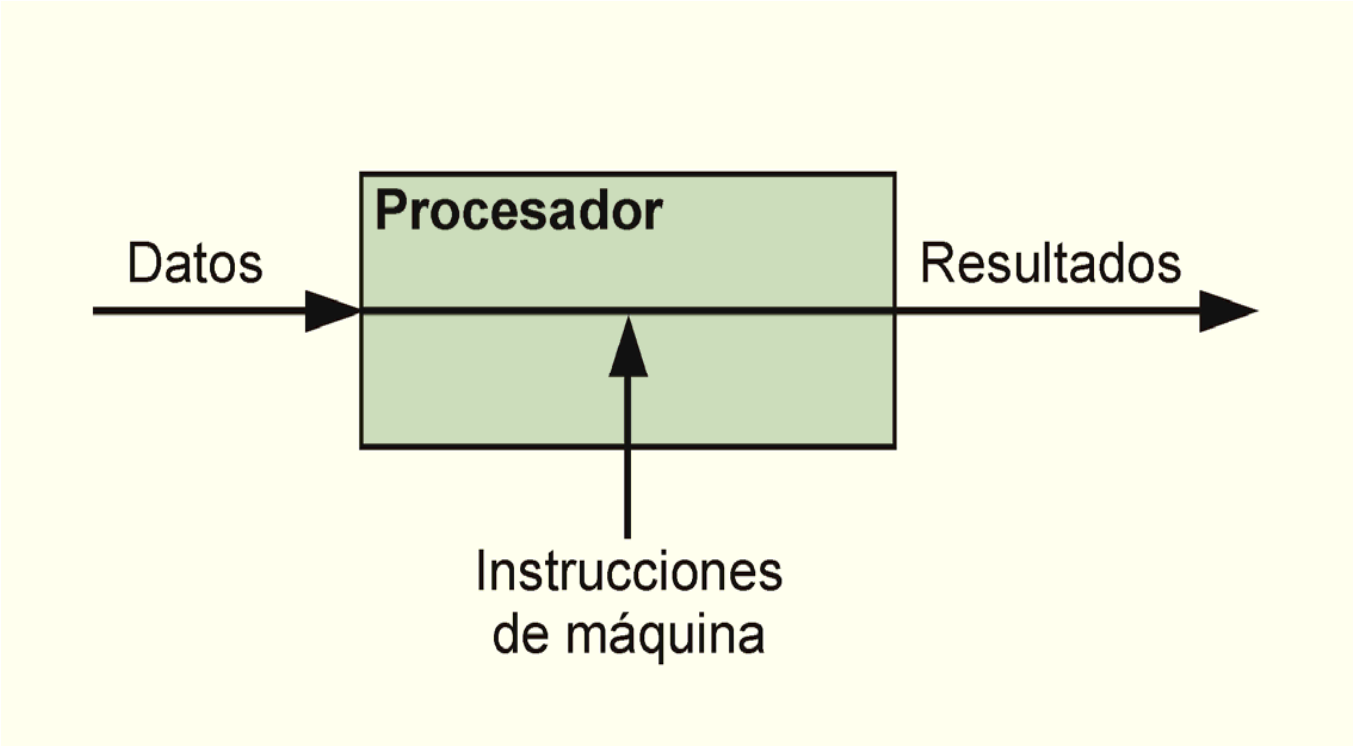
1.2 Soporte HW

subsistemas HW claves para el SO

- El SO está en contacto con el HW
  - La mayoría de las tareas del SO se impl. por SW
  - Pero algunas tareas
    - Son imposible de implementar sólo mediante SW
    - Serían muy lentas si no se implementan en HW
- Subsistemas HW de apoyo al SO
  - Soporte modo usuario/núcleo
  - Soporte de interrupciones
  - Acceso directo a memoria (DMA)
  - Reloj del computador
  - Jerarquía de memoria

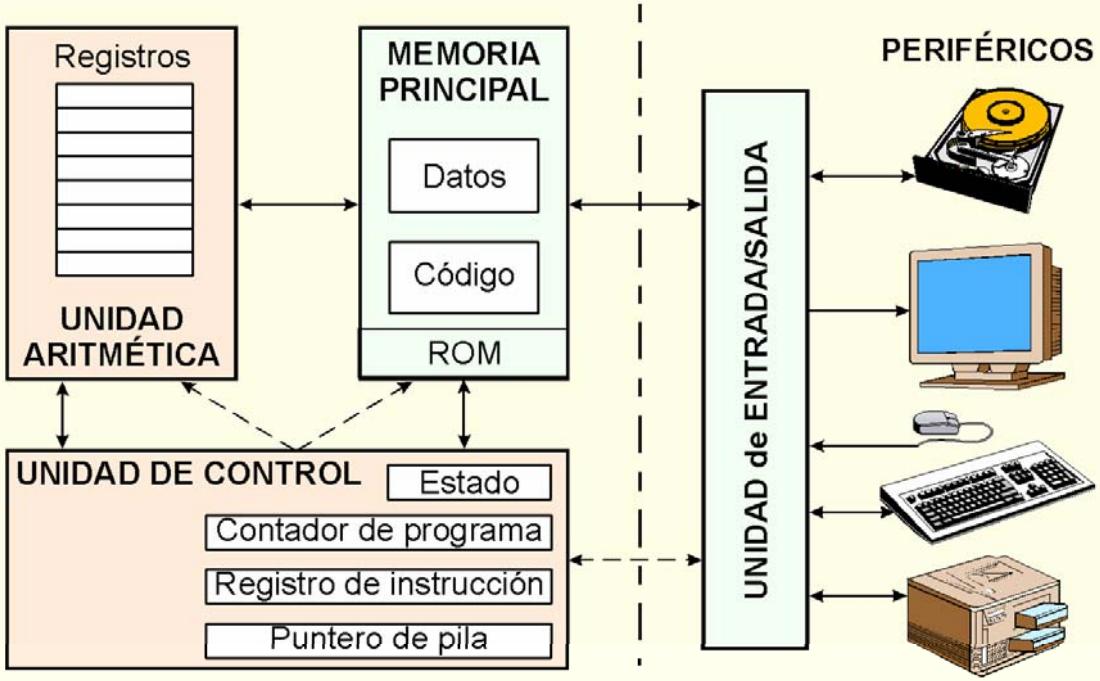
1.2 Soporte HW

esquema de funcionamiento del computador



1.2 Soporte HW

estructura del computador

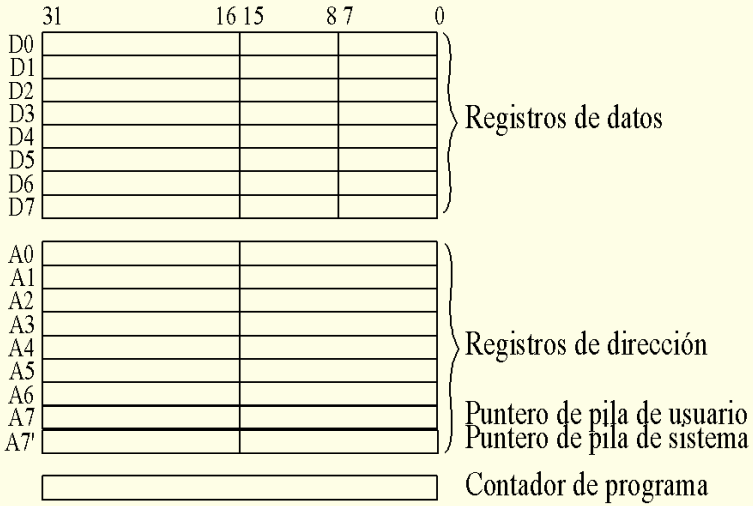


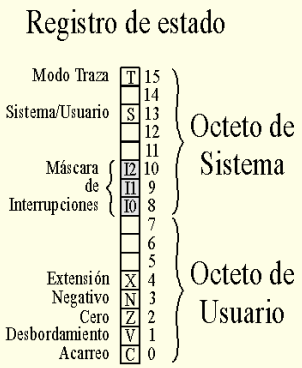
- Monoprocesador
- Multiprocesador
- Multicomputador

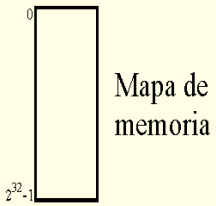
1.2 Soporte HW

modelo de programación del computador

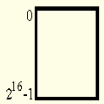
- Modelo de programación
- Juego de instrucciones (usuario y núcleo)







Mapa de memoria



Mapa de E/S

Juego de Instrucciones

1.2 Soporte HW

modo usuario/supervisor

Modelo de programación de usuario

Modelo de programación de núcleo

Fuente: Sistemas Operativos. Una visión aplicada. Fig 1.4.

1.2 Soporte HW

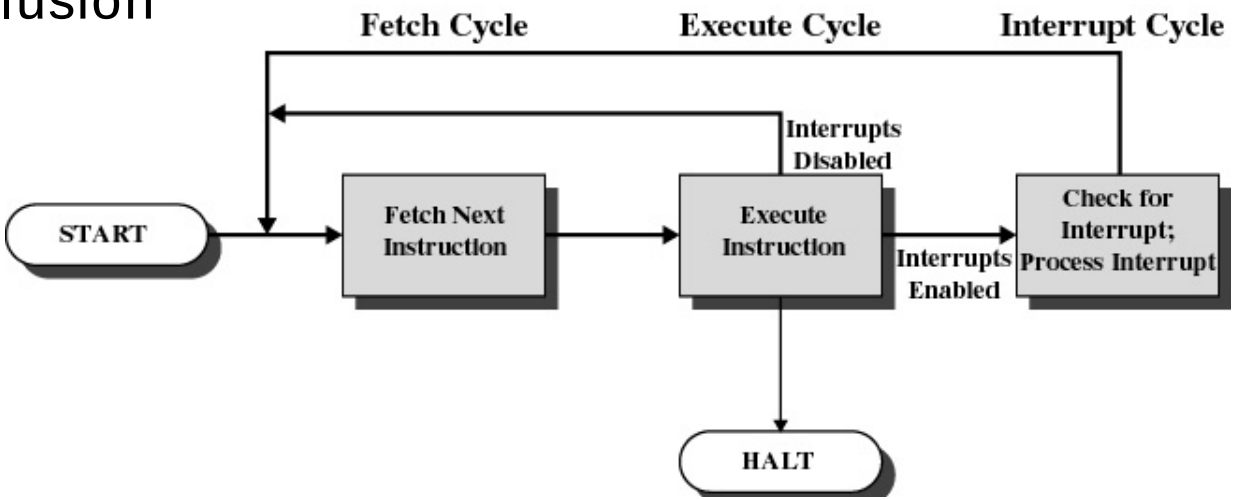
modo usuario/supervisor

- Dos niveles de ejecución. Flag en S.Reg.
  - Usuario:
    - Disponible un subconjunto de las instrucciones máquina
    - Algunos registros o partes de registros inaccesibles
    - Zonas del mapa de memoria prohibidas
    - Mapa de E/S (puertos de E/S) prohibidos
  - Supervisor
    - Sin ningún tipo de restricciones. Acceso total al HW.
- Dos modos de ejecución
  - El SO se ejecuta en modo núcleo
  - Los procesos de usuario se ejecutan en modo usuario
    - Cuando un proceso de usuario llama a una syscall se pasa a modo núcleo ya que la syscall pertenece al SO.

1.2 Soporte HW

interrupciones

- Mecanismo de transferencia de control: *interrupción* de la secuencia normal de ejecución
  - El ciclo de instrucción básico se modifica para su inclusión



- Mejora la eficiencia del procesamiento:
  - Permite al procesador ejecutar otras instrucciones mientras una operación de entrada/salida está en marcha (frente a E/S por programa)

1.2 Soporte HW

- Si llega una interrupción y está habilitada
  - Se espera a completar la inst. maquina en curso
  - Ejecutamos ciclo de aceptación de interrupción

HW
- Salvar PC y SR
- Se pasa a ejecución en modo núcleo
- Cargar en PC la dir. de la rutina de tratamiento de int. (RTI)
  - Esa dirección se obtiene de una tabla de vectores de int.
  - La tabla se indexa con el vector proporcionado por el agente que interrumpe

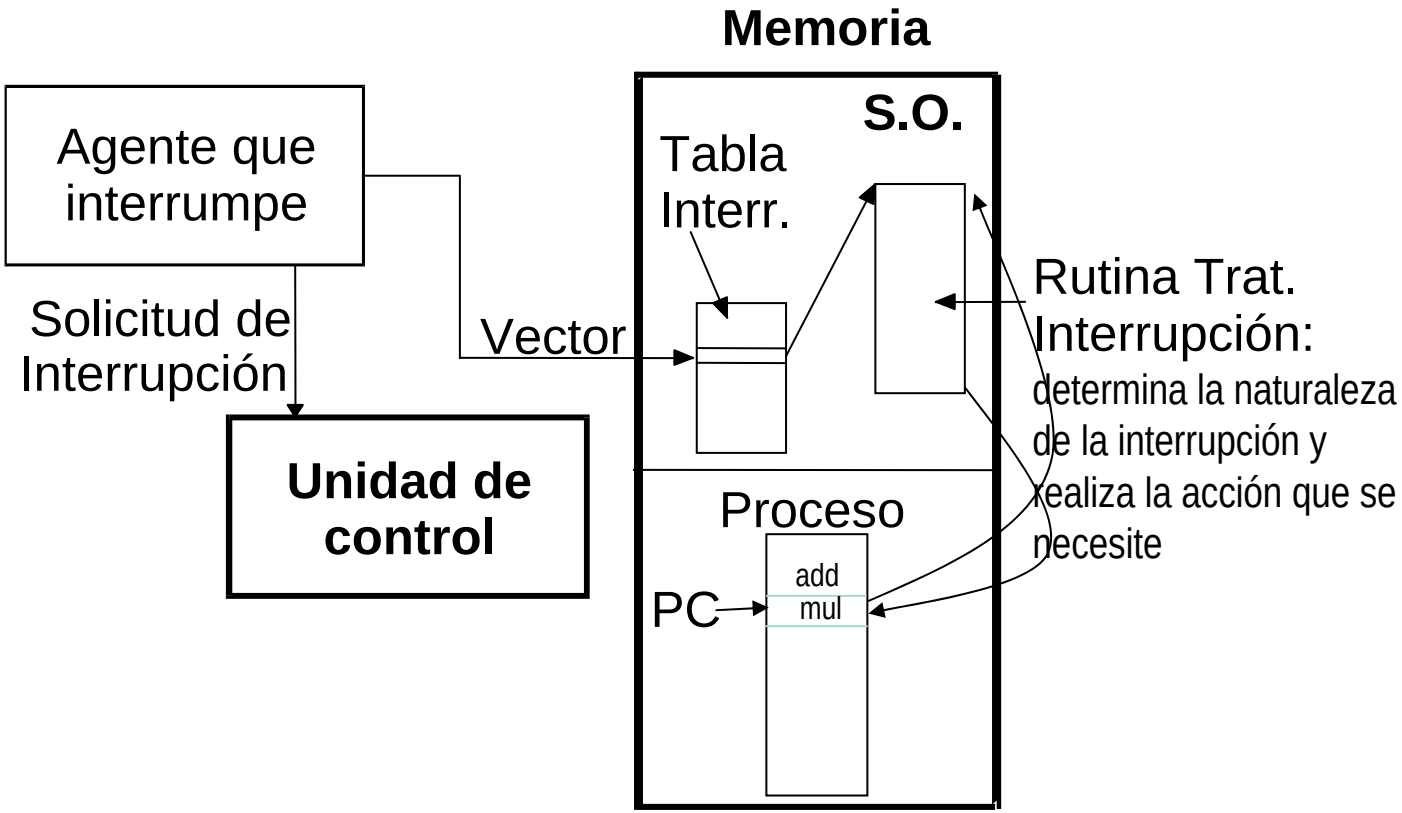
SW
- La RTI salva el resto de registros (estado del procesador)
- La RTI se ejecuta

HW
- Recuperamos PC y SR (así volvemos a modo usuario y el proceso interrumpido sigue su ejecución)

1.2 Soporte HW

interrupciones

- Las RTIs y la tabla de vectores son parte del SO



Fuente: Sistemas Operativos. Una visión aplicada. Fig 1.6.

1.2 Soporte HW

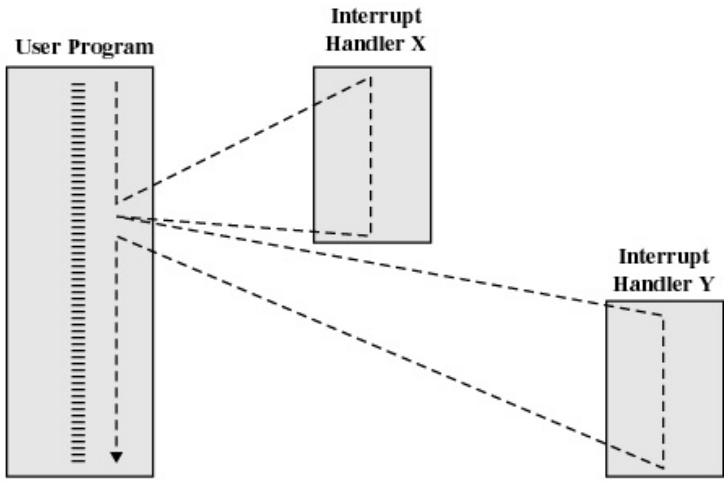
clasificación de las interrupciones

- Clasificación de las interrupciones
  - Int. SW o SVC (llamadas al supervisor)
    - Son realmente llamadas a syscall (al SO) ("trap" o "int")
    - No se pueden hacer por "call" (llamada a subrutina)
      - Hay que pasar a modo núcleo y el proceso no tiene acceso a las direcciones del espacio de mem. del SO
    - El número de vector de int. lo proporciona la instrucción
  - Int HW: activan una patilla del micro (int)
    - De E/S: controladores de periféricos, DMA, ...
    - De reset: no enmascarable. Una patilla diferente (reset)
    - De reloj: Permite que el SO se ejecute periódicamente
  - Excepciones: generada dentro del procesador
    - SW: división por cero, overflow, CO ilegal, fallo de página,...
    - HW: sobrecalentamiento, pico de tensión, paridad
    - El número de vector de int. lo proporciona el procesador

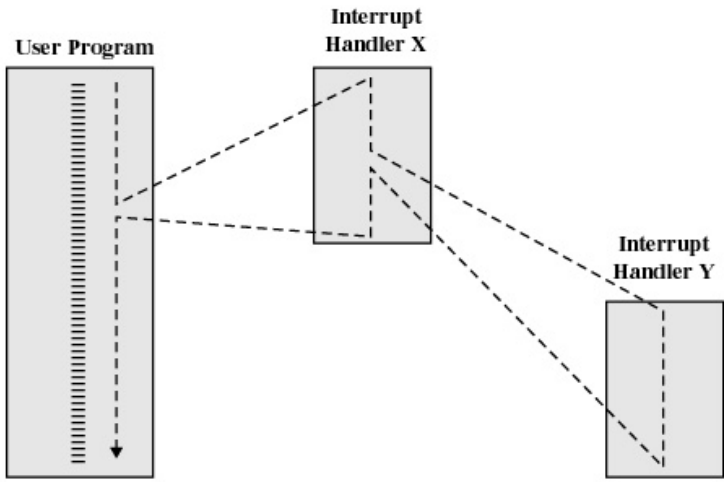
1.2 Soporte HW

interrupciones múltiples

- Orden secuencial:
  - Deshabilitar las interrupciones de manera que el procesador pueda completar la tarea
  - La interrupción queda pendiente hasta que el procesador permite interrupciones
  - Cuando acaba la rutina de tratamiento de la interrupción (RTI), el procesador comprueba si hay interrupciones pendientes
- Múltiples prioridades
  - Las interrupciones de mayor prioridad:
    - hacen esperar a las interrupciones de menor prioridad
    - interrumpen a las RTI de interrupciones de menor prioridad
  - p.e.: cuando llega un dato de una línea de comunicación, éste debe ser absorbido rápidamente para dejar sitio a más datos



(a) Sequential interrupt processing

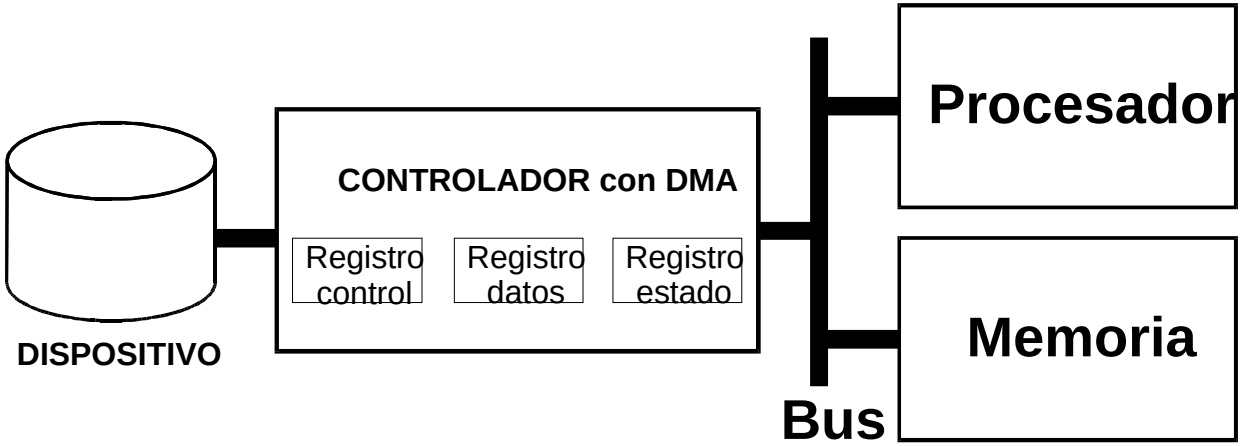


(b) Nested interrupt processing

1.2 Soporte HW

acceso directo a memoria (DMA)

- Permite máxima concurrencia CPU-E/S
- Evolución:
  - E/S programada: concurrencia nula
    - El procesador se encarga de toda la transferencia
  - E/S por interrupciones: concurrencia media
    - El procesador evita la espera activa del periférico
  - E/S por DMA: concurrencia máxima
    - El procesador encarga la operación E/S al DMA
    - CPU y DMA trabajan en paralelo: concurrencia CPU-E/S
    - El SO se encargará de explotar esta concurrencia

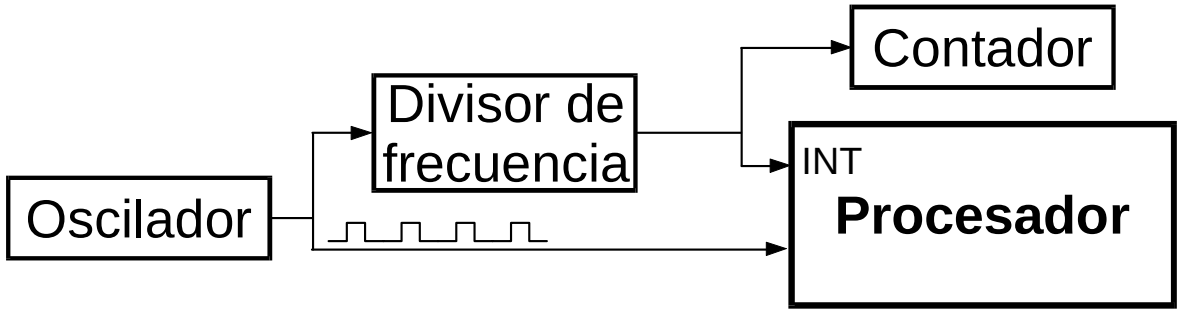


Fuente: Sistemas Operativos. Una visión aplicada. Fig 1.22.

1.2 Soporte HW

reloj del computador

- Tres visiones del reloj
  - Oscilador que gobierna las fases de las instrucciones máquina (CLK)
  - Generador de int. Periódicas (tics)
    - El SO se ejecuta periódicamente.
    - Se puede evitar que un proceso monopolice el procesador
  - Contador: actualiza fecha y hora (RTC)
    - Un contador de segundos desde una fecha determinada
    - En Unix se cuenta desde el 1/1/70. Un contador de 32 bits volverá a cero en el 2038.
    - Circuito dedicado permanetemente alimentado (SO en sist. Antiguos)



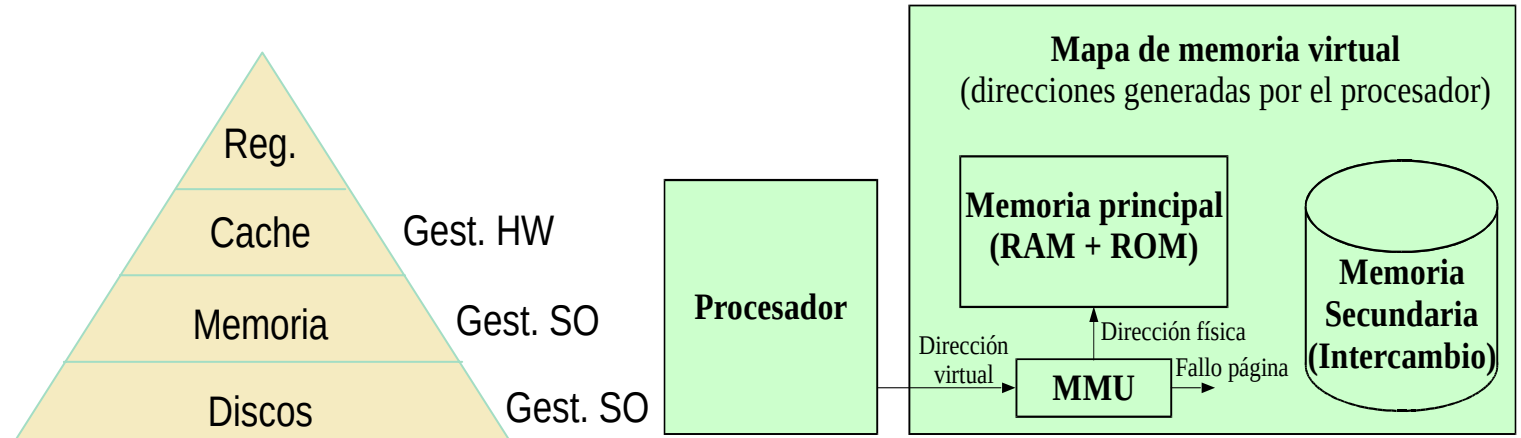
Fuente: Sistemas Operativos. Una visión aplicada. Fig 1.7.

1.2 Soporte HW

jerarquía de memoria

Jerarquía de memoria

- Bloques de datos replicados en varios niveles de la jerarquía
- Objetivo: simular una memoria grande, rápida y barata
- La CPU genera direcciones virtuales
  - Parte del mapa de memoria está en Mem. y parte en HD
  - La MMU traduce las dir. virtuales a dir. físicas
  - Fallo de página (trap) cuando la dir. no está en Mem. Principal
  - El SO trae una página a Mem (posible reemplazo en Mem): ALTO COSTE TEMPORAL



Fuente: Sistemas Operativos. Una visión aplicada. Fig 1.14.

1.3 Evolución

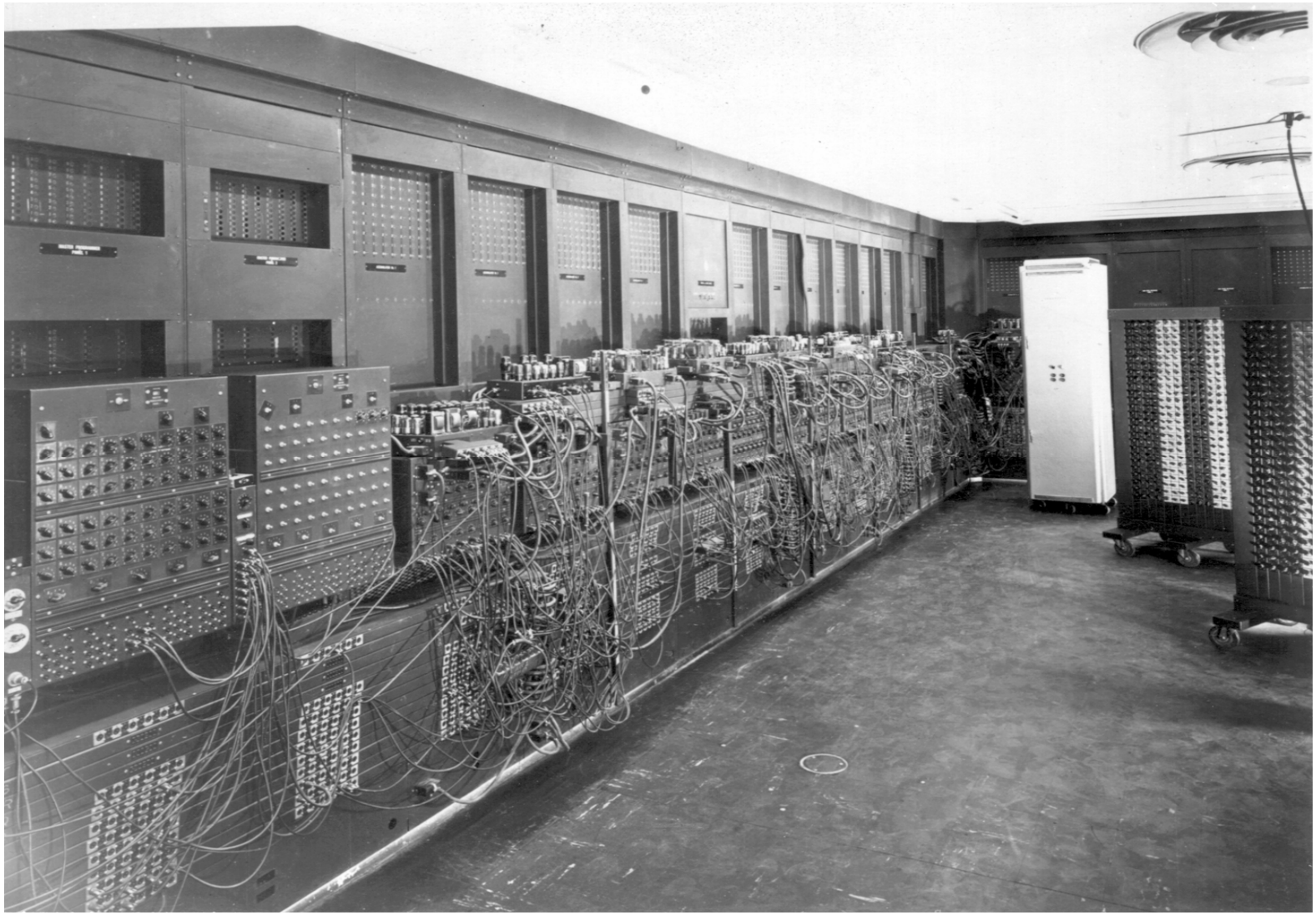
1.3 Evolución

Evolución de los Sist. Operativos

- procesamiento interactivo en serie ('45-'55)
- sistemas con procesamiento por lotes ('55-'65)
- multiprogramación por lotes, tiempo compartido, tiempo real ('65-'70)
- distribuidos, paralelos, de Internet

1.3 Evolución

’45-’55: procesamiento serie



1.3 Evolución

'45-'55: procesamiento serie

- Tecnología de válvulas
- No existe el S.O.
  - Las máquinas ejecutan desde una consola con luces e interruptores
  - El programador:
    - Reserva el lab. para usar la máquina: libro de reserva (Schedule tome)
    - Carga el programa en binario, ejecuta y analiza salida
    - Inconvenientes:
      - Sólo realizable para programas pequeños
      - Baja productividad: reservas 2h, pero acabas en 1.5horas
- Mejoras
  - Tarjetas perforadas, cintas (papel y magnét.), impresora.
  - Compiladores (Fortran y Cobol)

1.3 Evolución

'45-'55: procesamiento serie

- Los compiladores
  - Simplifican la programación
  - Complican la carga/ejecución de prog.
    - Cargar compilador, compilar, cargar ensamblador, ensamblar, cargar el binario, ejecutar, analizar salida.
    - Conclusión: tiempo de "preparación" muy largo. CPU ociosa
- Soluciones
  - Contratar operador.
    - El programador da las tarjetas al operador y vuelve otro día
    - Depuración más difícil: post-mortem
  - El operador agrupa tareas
    - Primero carga el compilador y compila todos los programas,
    - Luego carga el ensamblador y los ensambla todos
    - Luego los ejecuta todos.

1.3 Evolución

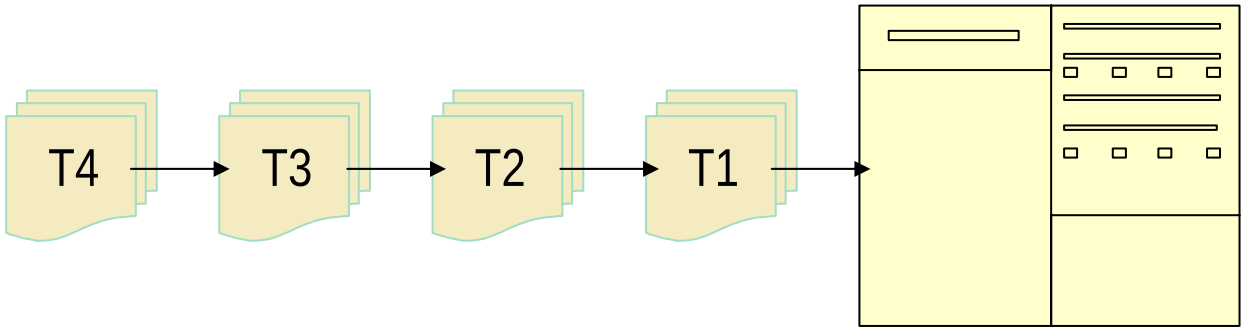
’55-’65: procesamiento por lotes

- Tecnología de transistores

1.3 Evolución

'55-'65: procesamiento por lotes

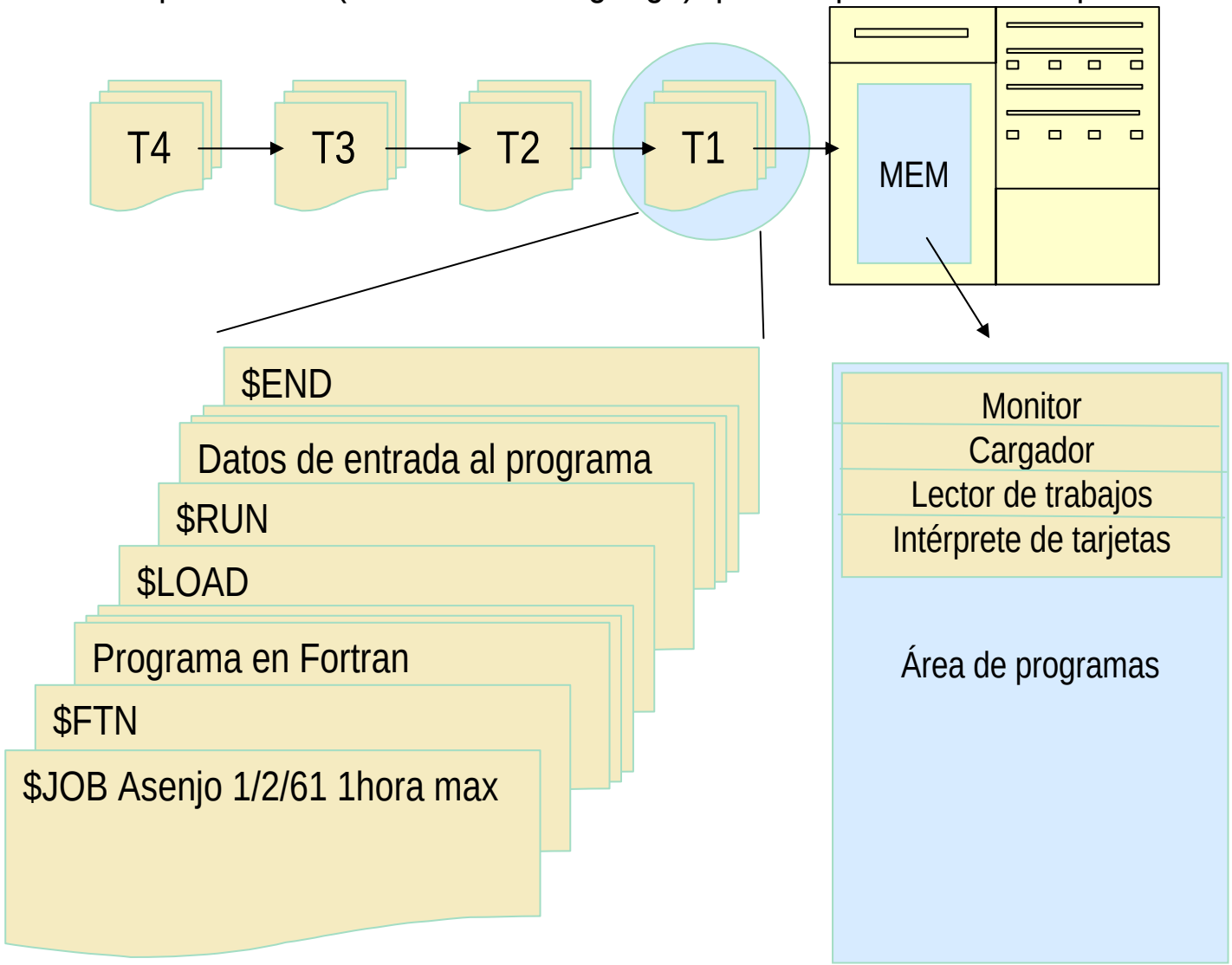
- Seguía existiendo una pérdida de Tcpu entre el final de un trabajo y el comienzo del siguiente
- Solución: sistemas por lotes
  - Objetivo: optimizan el flujo de trabajos, minimizando el tiempo empleado en retirar un trabajo y montar el siguiente
  - Primeros sistemas operativos
  - programa monitor que carga trabajos por lotes (batch):
    - Software que controla los programas que se ejecutan
    - Los lotes se ejecutan secuencialmente
    - El programa devuelve el control al monitor cuando termina
    - El monitor residente permanece en memoria principal y está listo para ejecutarse
  - Precursor de los ficheros .bat y scripts
  - Facilitan al usuario paquetes de rutinas de E/S



1.3 Evolución

'55-'65: procesamiento por lotes

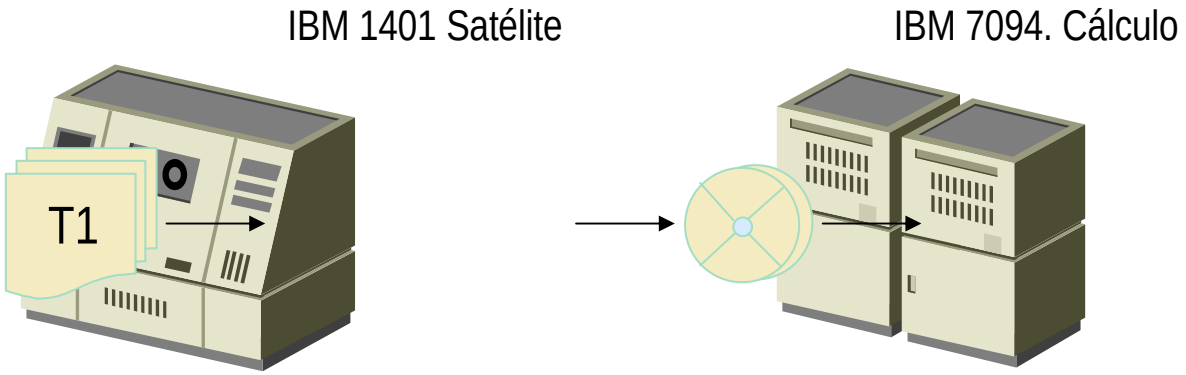
- Interpretan JCL (Job Control Language):qué compilador utilizar, qué datos...



1.3 Evolución

'55-'65: procesamiento por lotes

- Perdida de Tcpu en lectura tarjetas
  - Un programa de 1200 tarjetas tarda en leerse 60s y 4s en ejecutarse
    - CPU parada el 93.3% del tiempo
  - Solución: ejecución fuera de línea (off-line)
    - Pasar las tarjetas a cintas en otro computador dedicado (satélite) y ahora el computador de cómputo (principal) pierde menos tiempo en E/S ya que lee de cinta



1.3 Evolución

'55-'65: procesamiento por lotes

Características HW:

- Protección de memoria
  - Que no se altere el espacio de memoria del monitor
- Temporización
  - Evitar que un trabajo monopolice el sistema
- Interrupciones
  - Computadores más modernos ofrecen más flexibilidad
- Instrucciones privilegiadas
  - Instr. máquina que sólo debe ejecutar el monitor: E/S,...

1.3 Evolución

’65-’70

- Tecnología de CI.

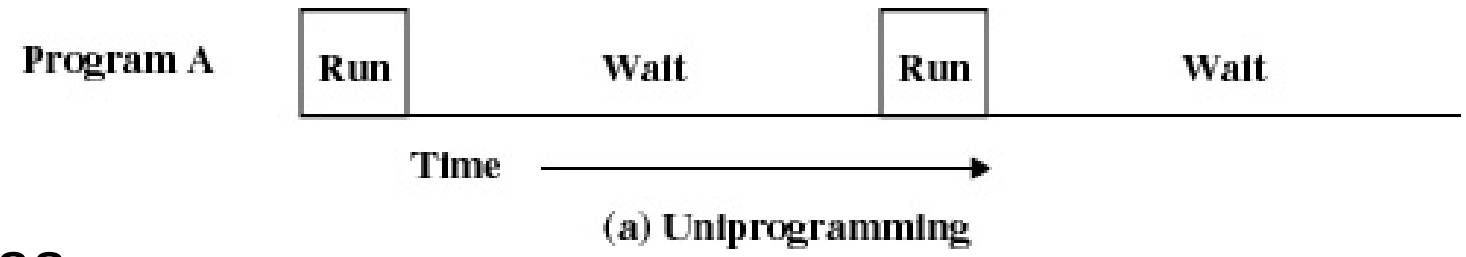
1.3 Evolución

’65-’70

Tecnología de CI.

- Nuevas tecnologías
  - Discos magnéticos
  - Gestión de interrupciones y Acceso Directo Mem (DMA)

- Objetivo SO: reducir tiempos de espera de E/S (monoprogramación)

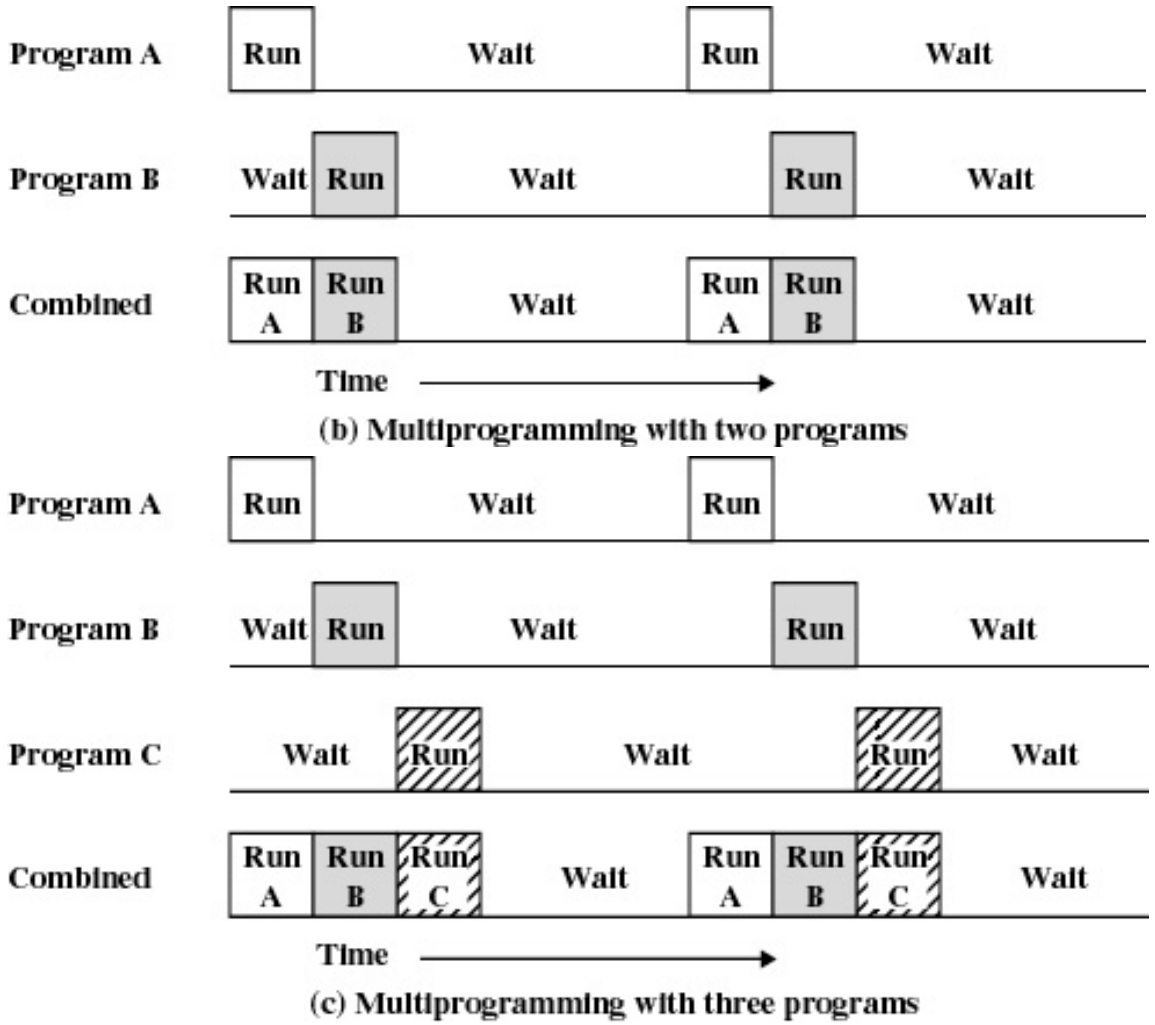


- Técnicas
  - Multiprogramación
  - Multiusuario o tiempo compartido (time-sharing)

1.3 Evolución

Multiprogramación

- Cuando un trabajo necesita esperar entrada/salida, el procesador puede cambiar a otro trabajo



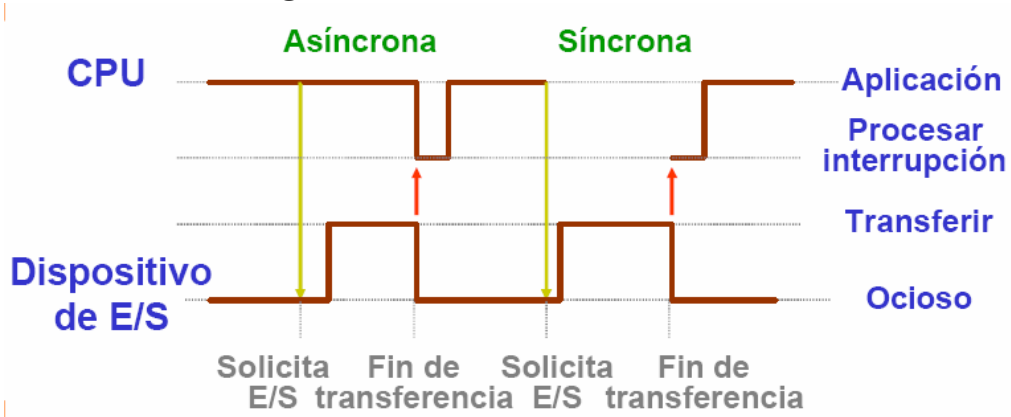
(b) Multiprogramming with two programs

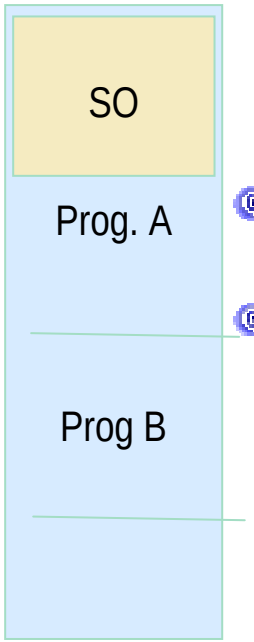
(c) Multiprogramming with three programs

1.3 Evolución

Multiprogramación

- El HW soporta las interrupciones de E/S y DMA: cuando un programa encarga un operación de E/S (síncrona), otro de los que están cargados en memoria usa la CPU





- Planificación: varios trabajos listos, hay que decidir cuál ejecutar
- Gestión de memoria. Mantener varios programas en memoria
  - Cada programa ocupa una porción de memoria
  - El SO, es uno de ellos
  - Hay que añadir mecanismos de protección para que un programa no lea/escriba posiciones de memoria de otro programa (y menos del SO)

1.3 Evolución

Multiprogramación. Ejemplo.

- Ejemplo: computador con 250 Mbytes disponibles (sin utilizar por el SO), un disco, un terminal y una impresora.

	TRABAJO1	TRABAJO2	TRABAJO3
Tipo de trabajo	Cálculo intenso	I/O	I/O
Duración	5 min.	15 min.	10 min.
Memoria	50 K	100 K	80 K
Necesita disco	No	No	SI
Necesita terminal	No	SI	No
Necesita imprimir	No	No	SI

1.3 Evolución

Multiprogramación. Ejemplo.

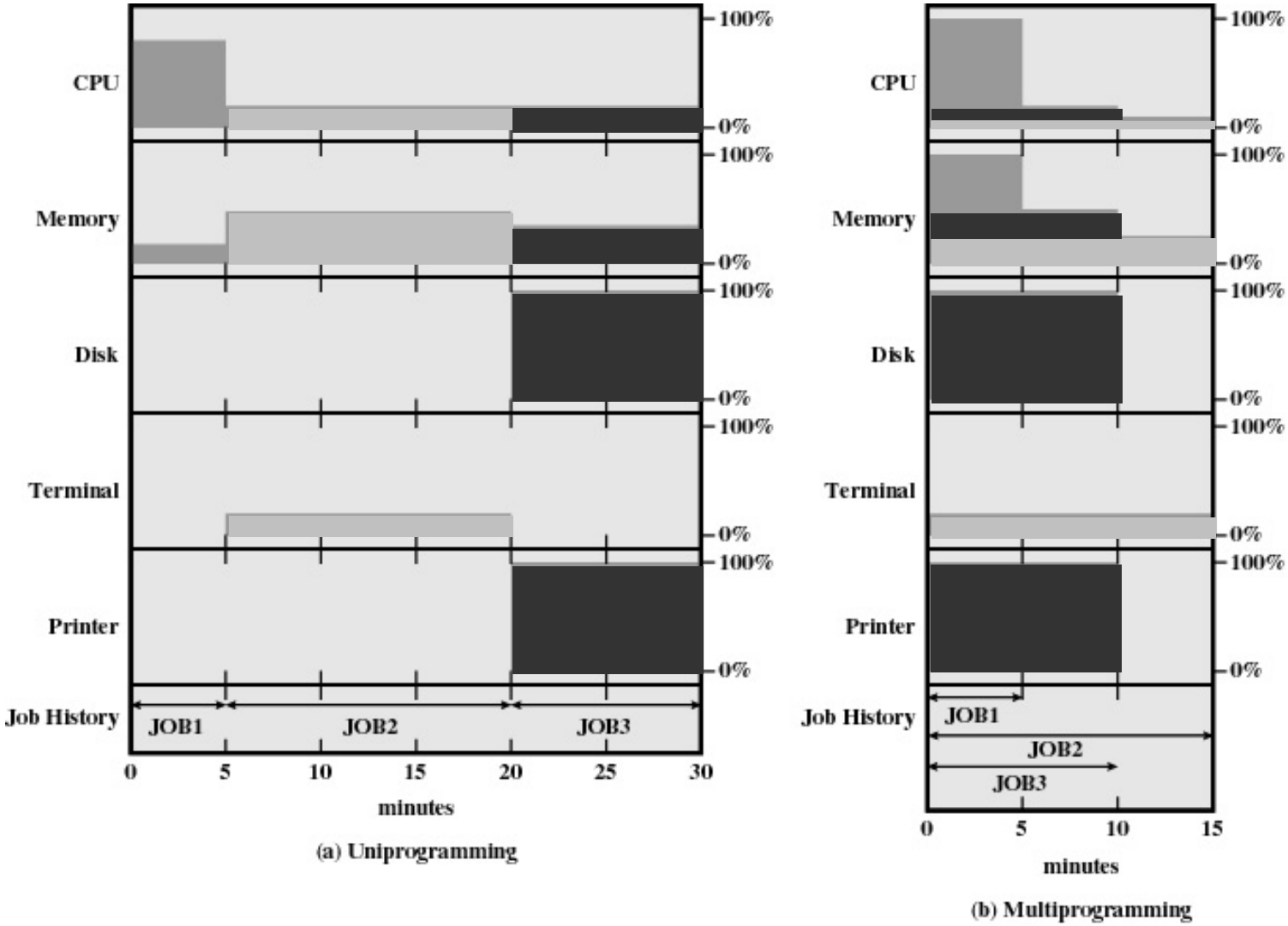


Figure 2.6 Utilization Histograms

1.3 Evolución

Multiprogramación. Ejemplo.

- Efectos de la multiprogramación

	Monoprogramación	Multiprogramación
Uso del Procesador	22%	43%
Uso de la Memoria	30%	67%
Uso del Disco	33%	67%
Uso de la Impresora	33%	67%
Tiempo Transcurrido	30 min.	15 min.
Productividad (ratio)	6 trabajos/hr	12 trabajos/hr
Tiempo Medio de Respuesta	18 min.	10 min.

1.3 Evolución

Tiempo compartido

- Consiste en utilizar multiprogramación para manejar varios programas interactivos
- Orientado a mejorar la interactividad
  - El procesamiento por lotes
    - Impide al usuario interactuar con el programa
    - Dificulta la depuración de programas (era dep. post-mortem)
  - Cuando aparece el terminal (teclado+monitor) tiene sentido interactuar con la máquina
    - Aparecen el interfaz de comando y el sistema de ficheros
    - El usuario actúa en función de resultados anteriores
    - Se necesita interactividad con tiempos de respuesta breve
    - Sin embargo: la máquina es cara para que la use un solo usuario
- Interactividad a bajo coste
  - Computador con varios terminales para varios usuarios. Puede comprarse entre varios.
  - Uso intensivo de la CPU: “multiplexa en el tiempo” la ejecución de los procesos

1.3 Evolución

Tiempo compartido

	Multiprogramación por Lotes	Tiempo Compartido
OBJETIVO PRINCIPAL	MAXIMIZAR EL USO DEL PROCESADOR	MINIMIZAR EL TIEMPO DE RESPUESTA
Fuente de directivas al sistema operativo	Comandos del lenguaje de control de trabajos proporcionados con el trabajo	Comandos introducidos desde el terminal

1.3 Evolución

Tiempo compartido

- Importancia del subsistema del SO para planificar (scheduler)
- Sistema multiusuario: autenticación y protección
- Los sistemas por lotes se siguen usando dentro de sistemas multitarea
  - Para trabajos grandes que no requieran intervención
- Un sistema multiusuario suele ser multitarea pero no tiene por qué
    - Ejemplo: programa de reservas aéreas: 1 sólo programa (= 1 sólo proceso) con varios terminales remotos

1.3 Evolución

SOs de Tiempo-Real

- Los STRs se suelen usar en aplicaciones especializadas, p.ej. sistemas de control, ...
- Garantizar la respuesta a sucesos físicos en intervalos de tiempo fijos.
- Problema: poder ejecutar las actividades del sistema con el fin de satisfacer todos los requisitos críticos de las mismas.
- Con el uso de aplicaciones multimedia sobre PC's, todos los SOs tienen requisitos de tiempo-real.

1.3 Evolución

Sistemas Distribuidos

- Un sistema distribuido es un sistema multicomputador pero que no tiene una memoria común, ni reloj.
- El objetivo básico es compartir recursos distribuidos: hardware impresoras,discos,...) o software (correo, IRC, Web, etc).
- Permiten un aumento de la fiabilidad del sistema. Si una parte falla, el resto del sistema puede seguir su ejecución, al menos, parcialmente.

1.3 Evolución

SOs Paralelos

- SOs para **sistemas multiprocesadores** (sistemas de computador con varios procesadores que comparten una única memoria y un reloj).
- Estos sistemas sustentan aplicaciones paralelas cuyo objetivo es obtener aumento de velocidad de tareas computacionalmente complejas, p. ej. Predicción meteorológica, simulaciones de explosiones atómicas, etc.
- **Multiprocesamiento Simétrico** (SMP) – todos los procesadores pueden ejecutar código del SO y de las aplicaciones.

1.3 Evolución

SOs de Internet

- Algunos vendedores mantienen que el PC es más complicado y caro de lo necesario: tiene el hardware/software necesario para mantener gran funcionalidad que nunca se utiliza.
- Proponen el computador de red que suministra la funcionalidad mínima para ejecutar un navegador de Internet, el cual cargará los applets que suministran la funcionalidad necesaria (SO y aplicaciones).

1.3 Evolución

SOs para WID

- WID = Wireless Information Devices (móviles, agendas electrónicas ...)

- Los WID son pequeños y móviles
- Es muy comprometido reducir un SO de PC para ellos: su SO debe gestionar muy bien sus escasos recursos; en especial: su limitada potencia eléctrica, de CPU y escasa memoria.

1.4 Funciones del SO

1.4 Funciones del SO

- Gestión de los recursos de la comput.
  - Gestión del Procesador: gestión de procesos
  - Gestión de la memoria
  - Gestión de E/S
- Ejecución de servicios para programas
  - Ofrece un conjunto de servicios: syscalls / API
  - Proporciona una máquina extendida
- Interfaz con el usuario
  - Proporciona un intérprete de mandatos (texto o GUI)
    - Espera una orden (comando o evento de ratón)
    - Analiza la orden y si es correcta la ejecuta
    - Concluida la orden vuelve a la espera

1.4 Funciones del SO

el SO como gestor de recursos

- Varios procesos comparten el HW
  - De uno o de varios usuarios
  - Los procesos compiten por los recursos
- El SO arbitra la asignación de recursos
  - Controla que recursos están ocupados y cuando se liberan (estructuras de datos)
  - Recursos físicos: CPU, MEM, E/S
  - Recursos lógicos: archivos, puertos de comun.
- Protección entre los usuarios
  - Confidencialidad de la información
  - Los procesos no deben interferirse
- Contabilidad
  - Monitorizar la cantidad de recursos usados

1.4 Funciones del SO

el SO como máquina extendida

- El SO ofrece servicios: syscalls
  - Permiten ejecutar de forma cómoda y protegida ciertas operaciones
    - La alternativa es complicar los programas y no tener protección
  - Tipos de servicios
    - Ejecución de programas: gestión de procesos
      - Crear, matar, comunicar, sincronizar, suspender, ...
    - Ordenes de E/S: abrir, leer y escribir en periféricos
    - Operaciones sobre archivos: un nivel de abstracción mayor
      - Crear, borrar, renombrar, abrir, escribir, leer y cerrar.
    - Detección y tratamiento de errores
      - El SO analiza todas las órdenes que recibe antes de realizarlas
      - Trata las distintas condiciones de error que detecte el HW
        - » de E/S (paridad), de ejecución (desbordamiento, CO ilegal, ...)

1.4 Funciones del SO

el SO como interfaz de usuario

- Intérprete de comandos: shell
  - Permite interactividad
  - Ejecuta un bucle infinito:
    - Espera orden, la ejecuta si es correcta, vuelve a la espera
- Dos tipos de intérpretes
  - Textual: ventana MS-DOS, terminal UNIX
    - Exige al usuario memorizar los comandos
    - Dependen del SO: type (MS-DOS) equivale a cat (UNIX)
  - GUI: Graphical User Interface
    - Entorno amigable de ventanas, iconos, menús, botones,...
- Archivos de mandatos (.bat o scripts)
  - Automatizan la ejecución de secuencias de comandos
  - Comandos de control de flujo: for, if, case,...

1.4 Funciones del SO

funciones de la interfaz de usuario

- Permitir acceder y manipular el sistema
  - Manipular archivos y directorios
  - Ejecución de programas
  - Herramientas de desarrollo de programas
  - Comunicación con otros sistemas
  - Información del estado del sistema
  - Configuración del interfaz y entorno
  - Intercambio de datos entre aplicaciones
  - Control de acceso
  - Otras utilidades: calculadora, reproductores,...
  - Sistema de ayuda interactivo
- Pregunta: ¿qué programas pertenecen al interfaz?

1.4 Funciones del SO

Control del SO

- El SO operativo multitarea debe evitar:
  - Las interferencias entre los procesos de usuario
  - Y entre los procesos y el SO.
- para llevarlo a cabo de forma segura necesitamos apoyo HW (modo dual de operación de la CPU, protección de memoria).
- Protegemos a los procesos y a al SO si controlamos la ejecución de instrucciones privilegiadas
  - Instrucciones privilegiadas son aquellas que pueden potencialmente dañar a otros procesos (i.e. operaciones de E/S, actualizar el reloj, desactivar interrupciones, manipular la MMU o el bit de protección)
  - En modo usuario, el intento de ejecutar una instrucción privilegiada produce una excepción

1.4 Funciones del SO

Control del SO

- ¿Cómo puede un proceso de usuario realizar una operación privilegiada, p. ej. una E/S, crear un proceso, obtener memoria, etc.?
- Solución: los servicios del SO se obtienen invocando un procedimiento del sistema a través de una **llamada al sistema**.
- Las llamadas al sistema se implementan a través de una interrupción software o, como algunos la denominan, trampa.

Los SOs operativos están controlados por interrupción: los manejadores de interrupciones son los puntos de entrada al SO. **El cambio de modo usuario a modo kernel se produce cuando se produce una interrupción.**

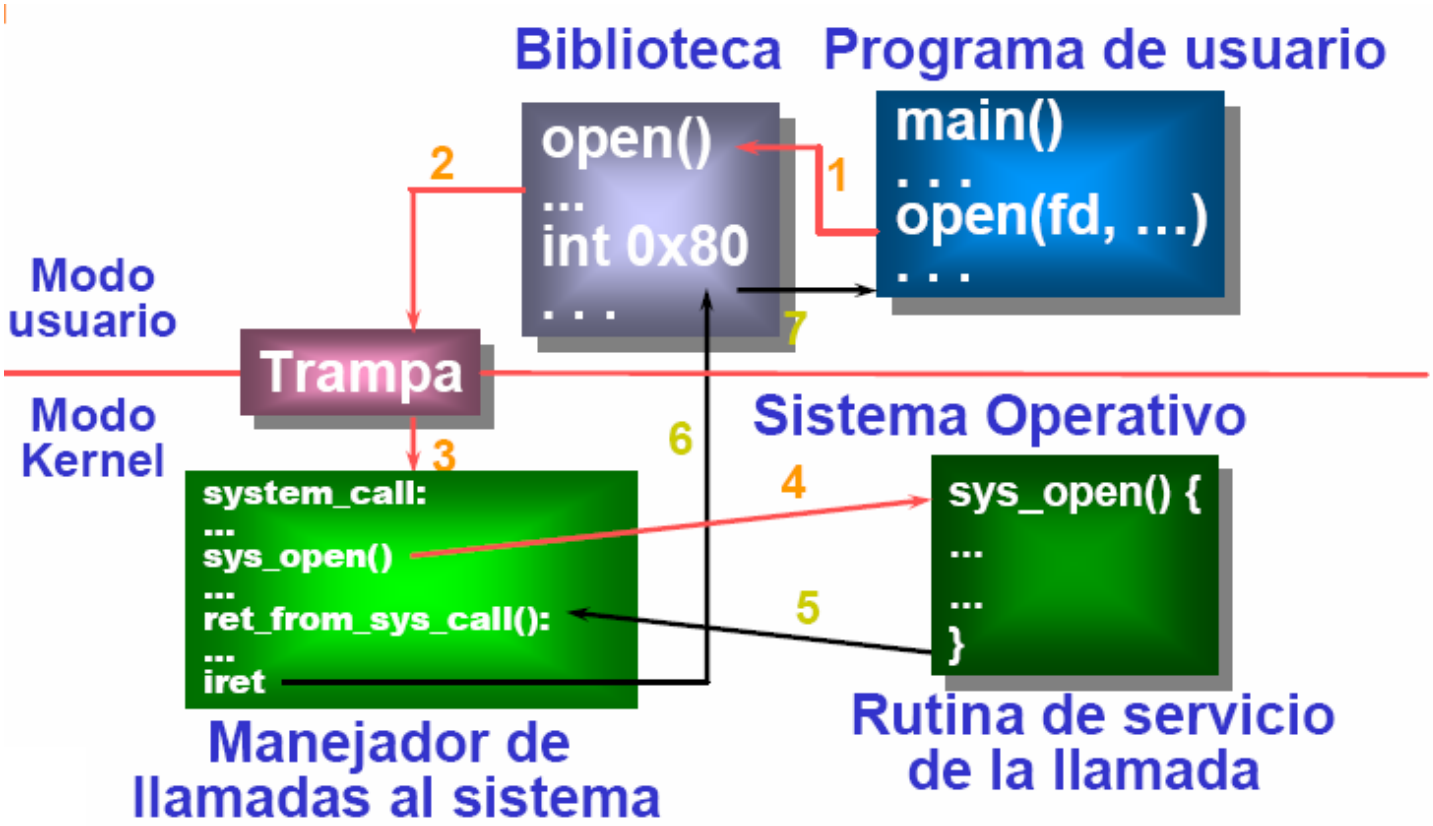
1.4 Funciones del SO

Realizando una llamada al sistema

1. Se genera una trampa a una posición específica del SO. P. ej. en Linux sobre Intelx86, con **`INT 0x80`**.
2. Este salto transfiere el control a la rutina de servicio de trampas del SO, y el bit de modo se activa (estamos en modo kernel).
3. El SO verifica que los parámetros son correctos y legales. Si es así, ejecuta la solicitud, y devuelve el control a la siguiente instrucción. Si no, devuelve error.

1.4 Funciones del SO

Realizando una llamada al sistema



1.4 Funciones del SO

Llamadas al sistema: paso de parámetros

- Para pasar parámetros se utiliza alguno de estos tres métodos:
  - Paso de parámetros en los registros. P.ej. Como en Linux:

```
load eax, valor
load bx, valor
....
INT 0X80
```

  - Almacenarlos en una tabla de memoria y pasar la dirección de la tabla como parámetro en un registro.
  - El programa empuja los parámetros en la pila, y el SO los recupera de la pila.

1.5 Estructuras del SO

en esta sección...

- Servicios (componentes) de un Sos
- Propiedades de una buena estructura
- Arquitecturas de SOs:
  - Monolítica
  - Capas
  - Microkernel

1.5 Estructuras del SO

Componentes de un SO

- Veremos los componentes de un SO y cómo estos se organizan.
- ¡Ten presente a partir de ahora que en los SOs, como en la vida real, nada es tan simple como parece! El diseño y la implementación del sistema no es tan clara en el mundo real como en los modelos.
- Los conceptos que veremos están presentes en todos los SOs, sin embargo, cada sistema los puede implementar de forma diferente.

1.5 Estructuras del SO

Servicios del SO

- Los servicios que suministra el SO a los procesos de aplicación (acceso a información permanente, a los dispositivos, suministrar memoria para ejecutarlos, garantiza el uso seguro de los recursos, suministra un interfaz de acceso a los servicios del SO) se suelen integrar en componentes.

1.5 Estructuras del SO

Gestión de procesos

- Podemos definir proceso como una instancia de un programa en ejecución.
- El SO es responsable de las siguientes actividades relacionadas con los procesos:
  - crear/destruir procesos,
  - suspender/reanudar procesos,
  - suministrar los mecanismos para
  - sincronizar y comunicar procesos.

1.5 Estructuras del SO

Gestión de memoria principal

- La memoria principal es el almacenamiento de acceso directo para la CPU y los dispositivos de E/S. Es volátil.
- El SO es responsable de:
  - asignar/desasignar memoria a los programas.
  - mantener la pista de la de memoria utilizada actualmente y quién la usa.
  - decidir cuanta memoria asignar a cada proceso, y cuando debe ser retirado de memoria un proceso.

1.5 Estructuras del SO

Gestión de archivos

- Un archivo es una colección de información con nombre. Es la entidad básica de almacenamiento persistente.
- El sistema de archivos suministra:
  - primitivas para manipular archivos y directorios: crear, borrar, leer, escribir, renombrar, ...
  - correspondencia entre archivos y su almacenamiento secundario.
- También, suministra servicios generales: backup, contabilidad y cuotas, etc.

1.5 Estructuras del SO

Gestión de Entradas/Salidas

- Los SOs ofrecen a los programas una interfaz estándar de los dispositivos, es decir, utilizan las mismas funciones independientemente del dispositivo al que accedan.
- Un **manejador de dispositivo** es un módulo que gestiona un tipo de dispositivo. El encapsula el conocimiento específico del dispositivo, p.ej., inicialización, interrupciones, lectura/escritura, etc.

1.5 Estructuras del SO

Sistema de protección

- Protección referencia al mecanismo para controlar los accesos de los programas a los recursos del sistema.
- El mecanismo de protección debe:
  - distinguir entre uso autorizado o no,
  - especificar que control se debe imponer, y
  - suministrar los medios para su aplicación.
- La protección suele ser un mecanismo general a todo el SO, es decir, no esta localizada en un único módulo.

1.5 Estructuras del SO

Intérprete de órdenes

- Programa o proceso que maneja la interpretación de órdenes del usuario desde un terminal, o archivo de órdenes, para acceder a los servicios del SO.
- Puede ser una parte estándar del SO, p. ej. El command.com de MS-DOS.
- O bien, un proceso no privilegiado que hace de interfaz con el usuario. Esto permite la sustitución de un interprete por otro. P. ej. en Unix tenemos csh, bash, ksh, etc.

1.5 Estructuras del SO

Estructura del SO

- Un SO consta de los elementos vistos anteriormente, entre los cuales existen muchas y complejas relaciones.
- Las preguntas importantes son:
  - ¿Cómo se organiza todo esto?
  - ¿cómo construir un sistema complejo que sea eficiente, fiable, extensible...?

1.5 Estructuras del SO

características deseables del SO

- Eficiencia
  - El SO debe ser lo más eficiente posible, pues consume muchos ciclos de CPU
  - Minimizar Tcpu ociosa y Trespuesta
  - Maximizar utilización de recursos y rendimiento
- Fiabilidad
  - Debe estar libre de errores en cualquier situación
  - Tolerancia a fallos:
    - el SO debe responder de forma predecible a condiciones de error, incluidos fallos HW.
    - El SO debe protegerse activamente a sí mismo y a los usuarios de acciones accidentales o malintencionadas.

1.5 Estructuras del SO

características deseables del SO

- Facilidad de mantenimiento:
  - Coste de actualización. **Extensibilidad**: la aparición constante de nuevo hardware y de nuevos tipos de aplicaciones, exigen al SO la adición de nueva funcionalidad. En lugar de construir un nuevo SO cada vez, se pretende construir un sistema que pueda extenderse, es decir, su funcionalidad pueda variar o crecer, de una forma sencilla. Por ejemplo, que un sistema tradicional soporte aplicaciones de tiempo real, que soporte nuevos sistemas de archivos, etc.
  - Coste de corrección
  - Construcción modular con interfaz bien definido
  - Bien documentado
- Tamaño reducido
- Portabilidad a varias arquitecturas

1.5 Estructuras del SO

implementación del SO

- El SO es un SW de gran complejidad
- Dos grupos de implementaciones:
  - Monolíticos
  - Estructurados:
    - por capas,
    - microkernel.
- Monolíticos (MSDOS, UNIX):
  - Los componentes del SO se ejecutan en modo kernel; la relación entre ellos es compleja.
  - El modelo de obtención de servicios es la llamada a procedimiento.
  - Son difíciles de comprender, ya que son un único programa (cientos de MB), por tanto, difíciles de modificar y mantener.
  - No fiables: un fallo de algún módulo puede provocar la “caída” del sistema.
  - Por esto, los diseñadores buscan mejores formas de estructurar un SO para simplificar su diseño, construcción, depuración, ampliación y mejora de sus funciones.

1.5 Estructuras del SO

implementación del SO

- Estructurados: sistemas por capas (OS/2):
  - Implementado como una serie de capas; cada una es una máquina abstracta para la capa superior.
  - Por modularidad, las capas se seleccionan para que cada una utilice sólo funciones de las capas inferiores.
  - Los sistemas de capas son jerárquicos pero los reales son más complejos, p. ej., el sistema de archivos podría ser un proceso en la capa de memoria virtual.
  - Problemas: existe sobrecarga de comunicaciones entre las distintas capas. A menudo, los sistemas se modelan con esta estructura pero no se construyen así.

1.5 Estructuras del SO

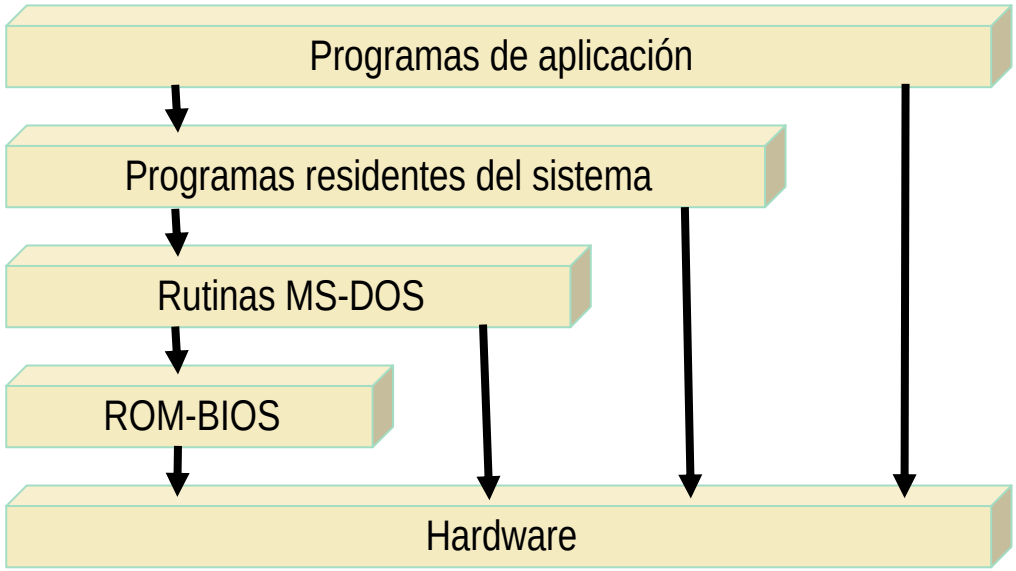
implementación del SO

- Estructurados: modelo cliente-servidor o microkernel (WindowsXP)
  - El núcleo del SO (microkernel) sólo implementa las funciones básicas que permiten la cración y comunicación de procesos
  - El resto de funciones se implementan como procesos de usuario (servidores o daemons)
  - Ventajas:
    - Más fiable: un posible error de un servicio del SO queda confinado en el espacio de direcciones del proceso que lo implementa
    - Es extensible y personalizable: podemos cambiar un servicio del SO cambiando el proceso que lo implementa.

1.5 Estructuras del SO

ejemplos de implementaciones

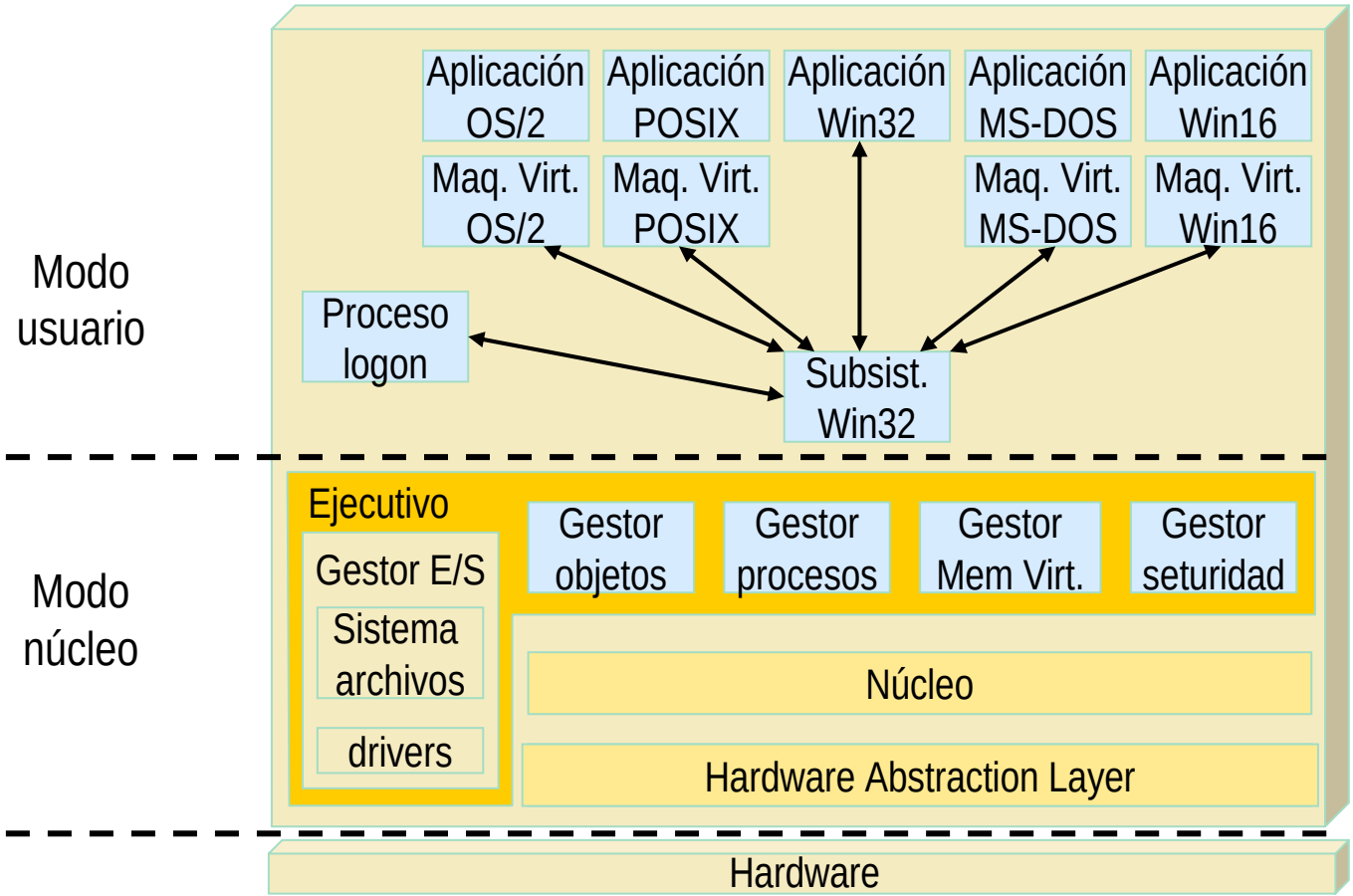
- Implementación de MS-DOS
  - Monolítico: fue creciendo poco a poco desordenadamente
  - Inicialmente orientado a ocupar poco espacio
  - Las aplicaciones tienen acceso directo al HW
    - El SO es vulnerable a programas imperfectos
    - A cambio es rápido y sencillo acceder al HW



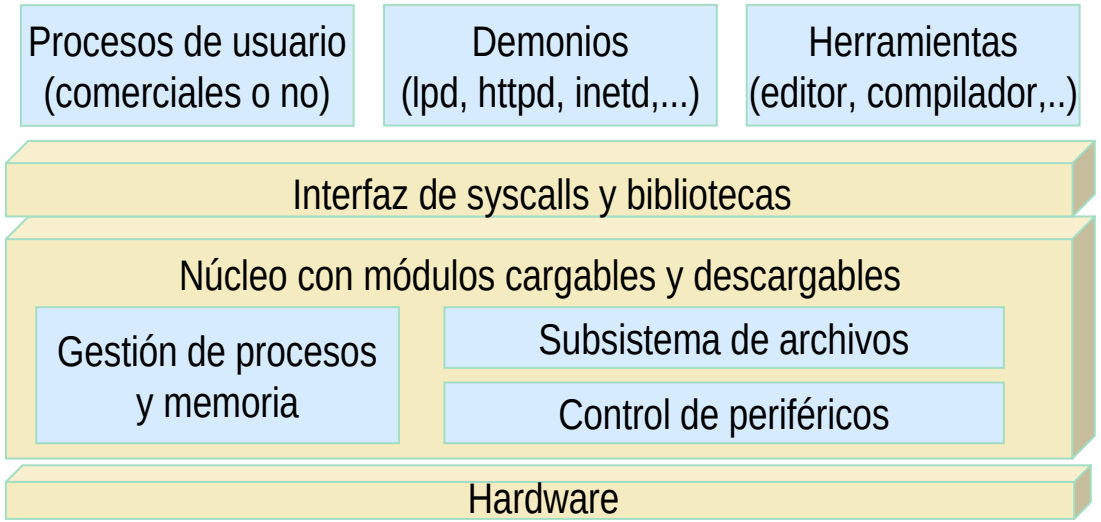
1.5 Estructuras del SO

ejemplos de implementaciones

- Implementación de Windows XP
  - Tipo microkernel (cliente servidor)



1.5 Estructuras del SO

ejemplos de implementaciones

- Implementación de Linux
  - Núcleo monolítico en aras de la velocidad
  - Pero implementado modularmente
    - Organizado en módulos independientes cargables
    - Proporciona las funciones básicas
  - Las bibliotecas del sistema proporcionan el resto
    - Interfaz de syscalls POSIX
    - Funciones matemáticas, manejo de cadenas, ordenación, ...