1 INTRODUCCIÓN

1 ¿QUÉ ES UN SISTEMA OPERATIVO?

1.2 SISTEMAS SIMPLES

1.3 SISTEMAS BATCH MULTIPROGRAMADOS

1.4 SISTEMAS DE TIEMPO COMPARTIDO

1.5 SISTEMAS DE COMPUTACIÓN PERSONAL

1.6 SISTEMAS PARALELOS

1.7 SISTEMAS DISTRIBUIDOS

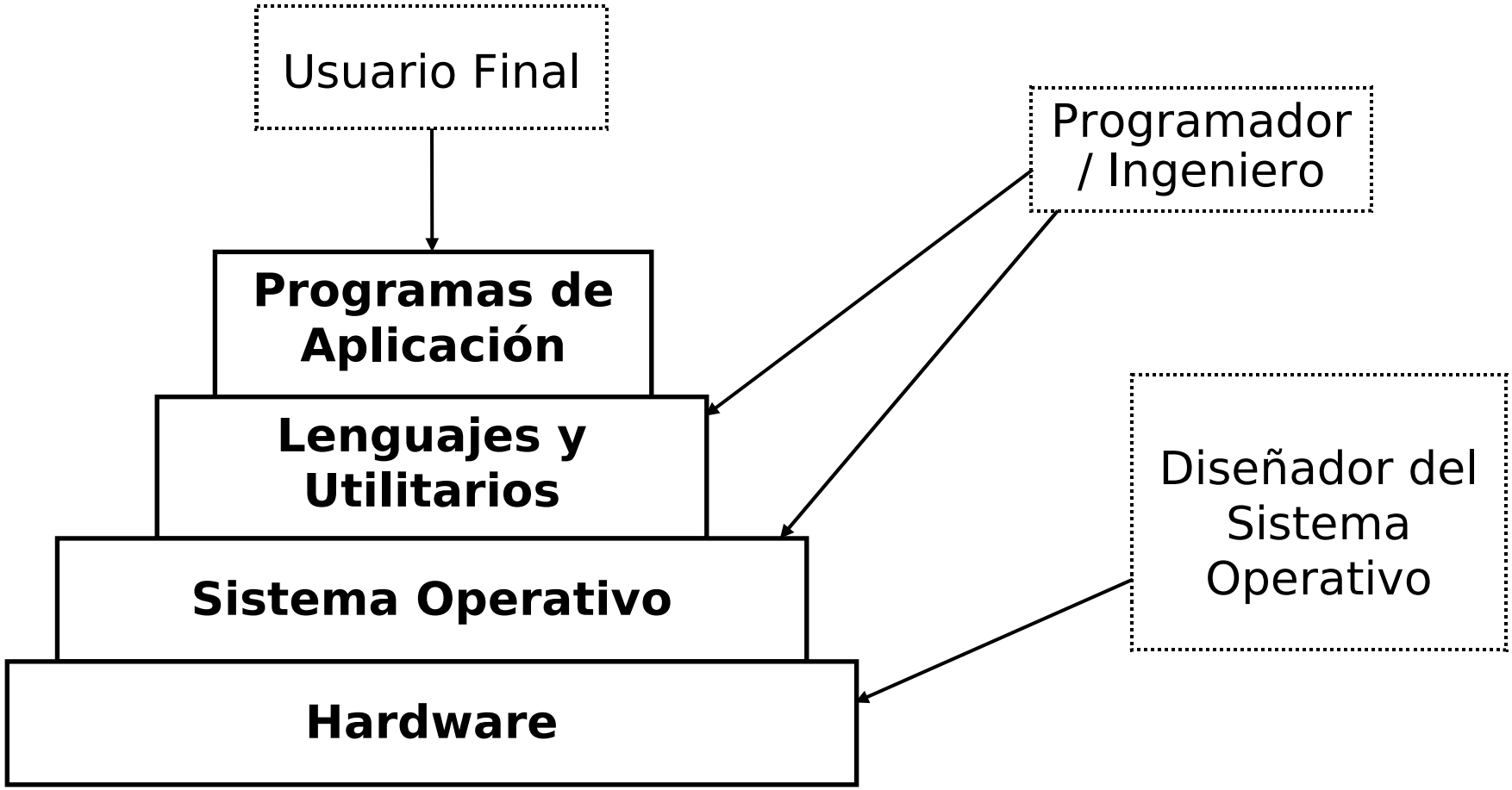
1.8 SISTEMAS DE TIEMPO REAL

1.9 AMBIENTES DE COMPUTACIÓN

DEFINICIONES Y CONCEPTOS BÁSICOS

1.1 ¿QUÉ ES UN SISTEMA OPERATIVO?

Componentes de un Sistema Computacional



1.1 ¿QUÉ ES UN SISTEMA OPERATIVO?

Componentes de un Sistema Computacional

1. **Hardware**: proporciona los recursos básicos de computación (CPU, memoria, dispositivos de E/S).
2. **Sistema Operativo**: controla y coordina el uso del hardware entre varios programas de aplicación para diferentes usuarios.
3. **Programas de Aplicación**: define la forma en que los recursos del sistema serán usados para resolver los problemas de índole computacional de los usuarios (compiladores, sistemas de base de datos, juegos de video, programas de negocio).
4. **Usuarios** (gente, máquinas, otros computadores).

1.1 ¿QUÉ ES UN SISTEMA OPERATIVO?

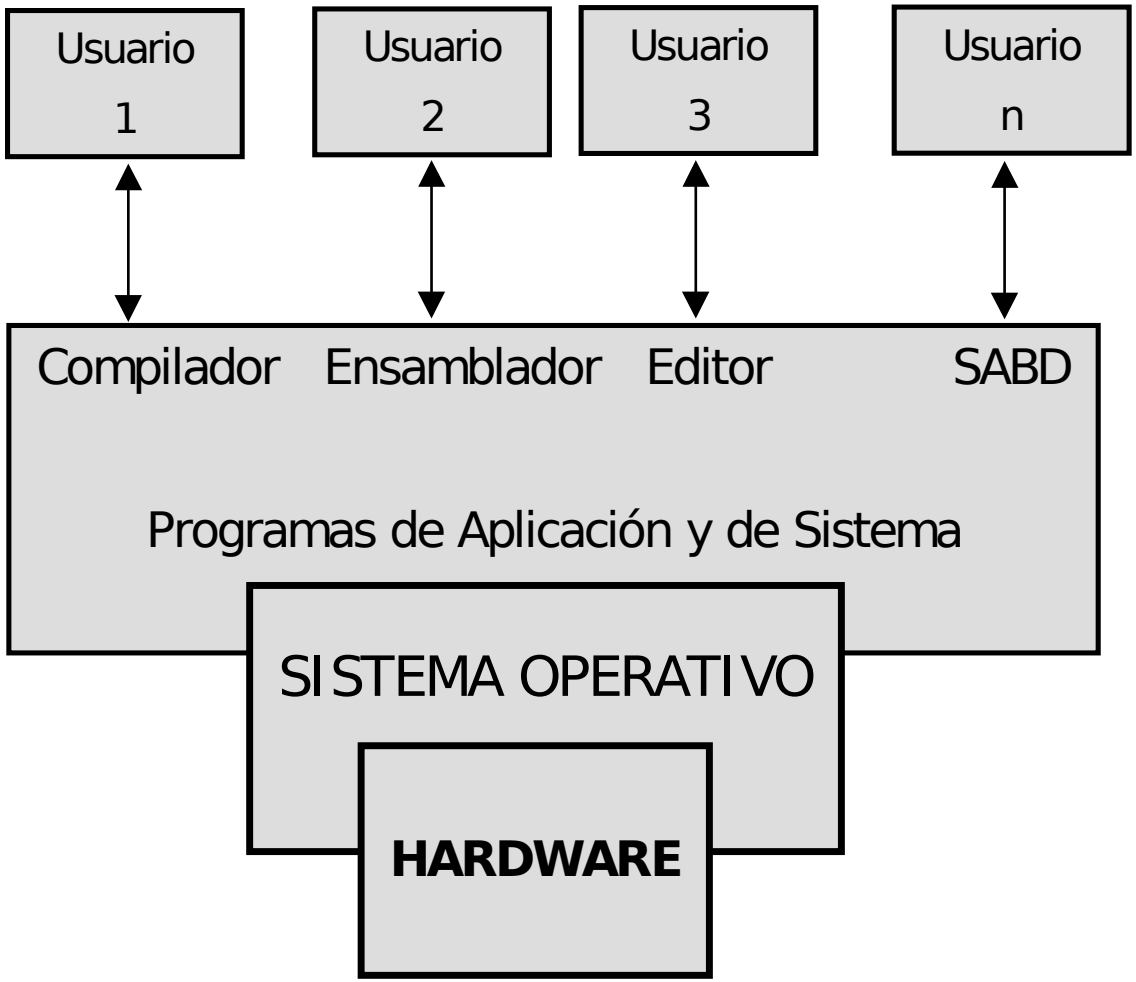
- Antiguamente, se definía a un sistema operativo como “el software que controla al hardware”.
- El panorama de los sistemas computacionales ha evolucionado significativamente, requiriéndose una definición más complicada.
- Las aplicaciones actualmente son diseñadas para ejecutarse de manera concurrente.

1.1 ¿QUÉ ES UN SISTEMA OPERATIVO?

- Es un programa que actúa como intermediario entre un usuario y el hardware del computador.
- El Objetivo primario de un Sistema Operativo (eficacia) es crear un entorno para:
  - La ejecución de programas de usuario y hacer que la resolución de problemas sea más fácil.
  - Hacer al sistema computacional conveniente de usar.
- El objetivo secundario es utilizar el hardware del computador de la mejor manera (eficiencia).

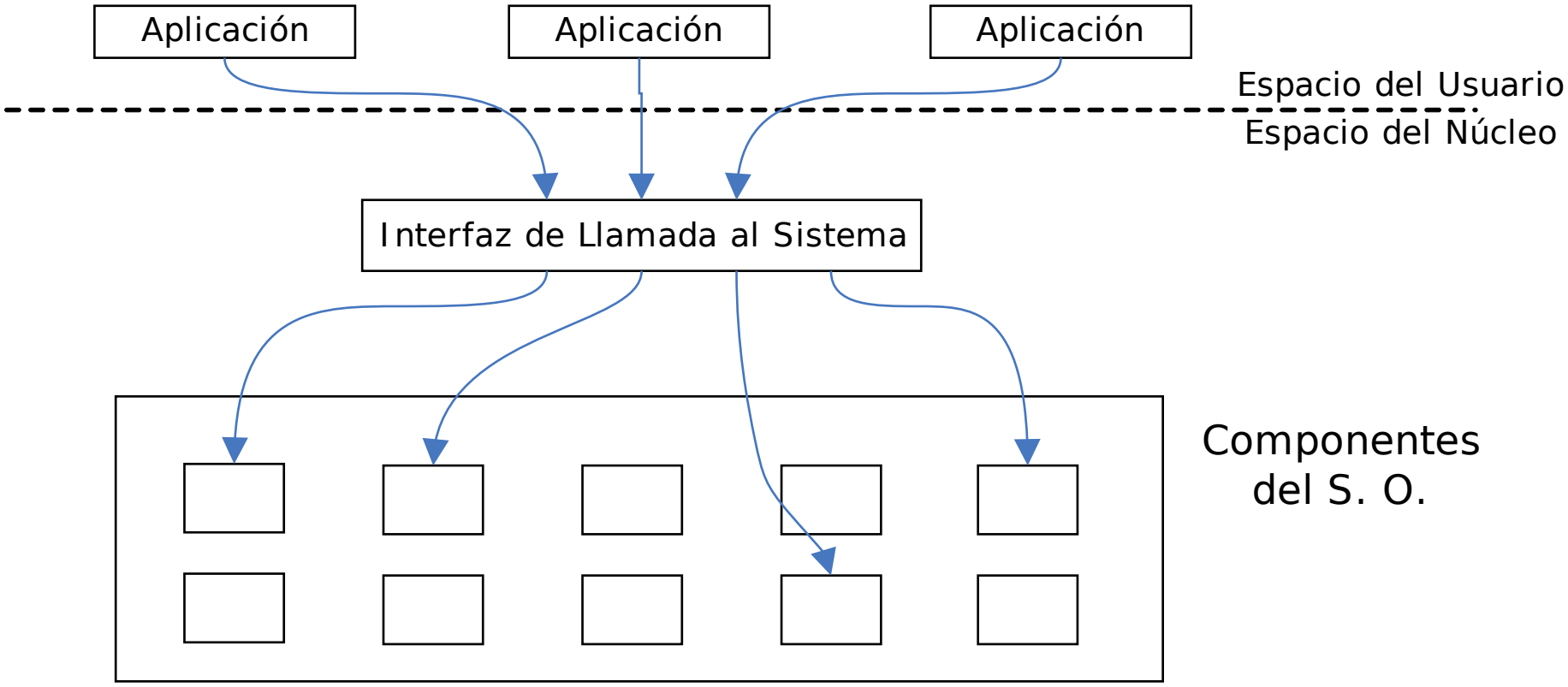
1.1 ¿QUÉ ES UN SISTEMA OPERATIVO?

Visión abstracta de los Componentes de un Sistema



1.1 ¿QUÉ ES UN SISTEMA OPERATIVO?

Interacción entre las aplicaciones y el sistema operativo



1.1 ¿QUÉ ES UN SISTEMA OPERATIVO?

Definiciones de Sistema Operativo

- **Asignador de Recursos**: administra y asigna recursos.
- **Programa de Control**: controla la ejecución de los programas de usuario y la operación de los dispositivos de E/S.
- **Kernel (núcleo)**: el único programa que se ejecuta todo el tiempo (todo el resto son programas de aplicación).

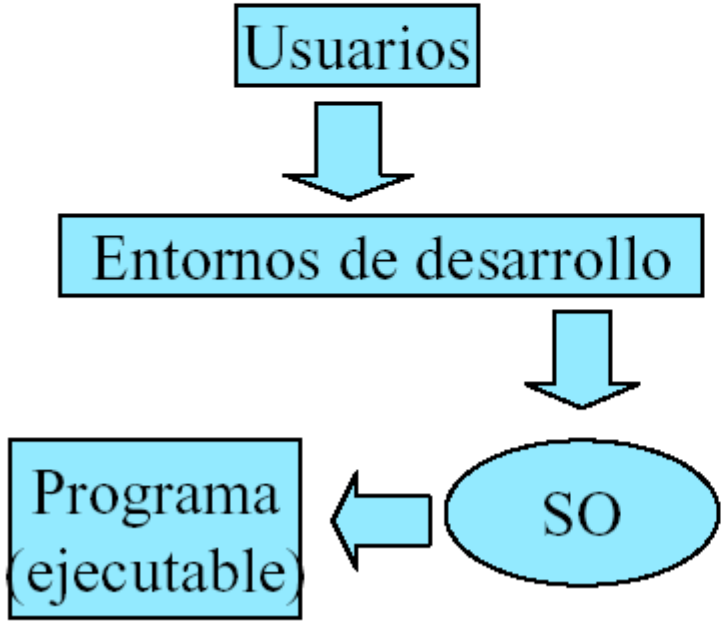
Otras consideraciones

- El SO es un programa como cualquier otro => ocupa tiempo de CPU
- EL SO dirige al procesador en el uso de los recursos y en el tiempo de ejecución del resto de los programas.
- Parte del SO reside en memoria. Contiene las funciones más importantes
- El resto de la memoria contiene otros programas.
- La distribución de la Memoria también es realizada por el SO, así como los dispositivos de I/O y Procesador.

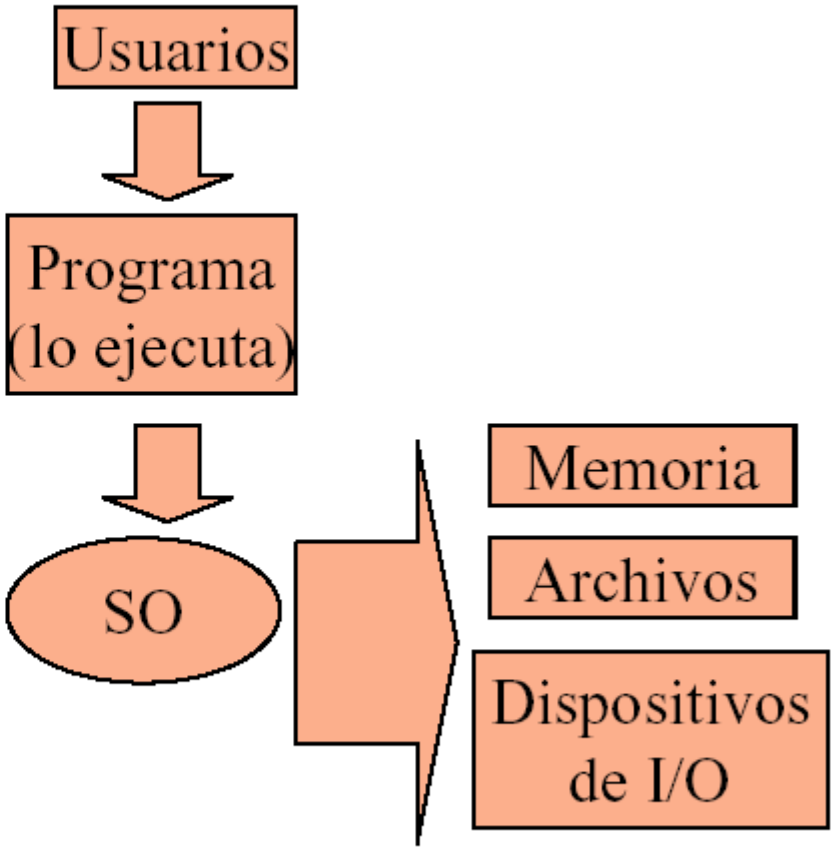
Usos Generales

Creación de Programas

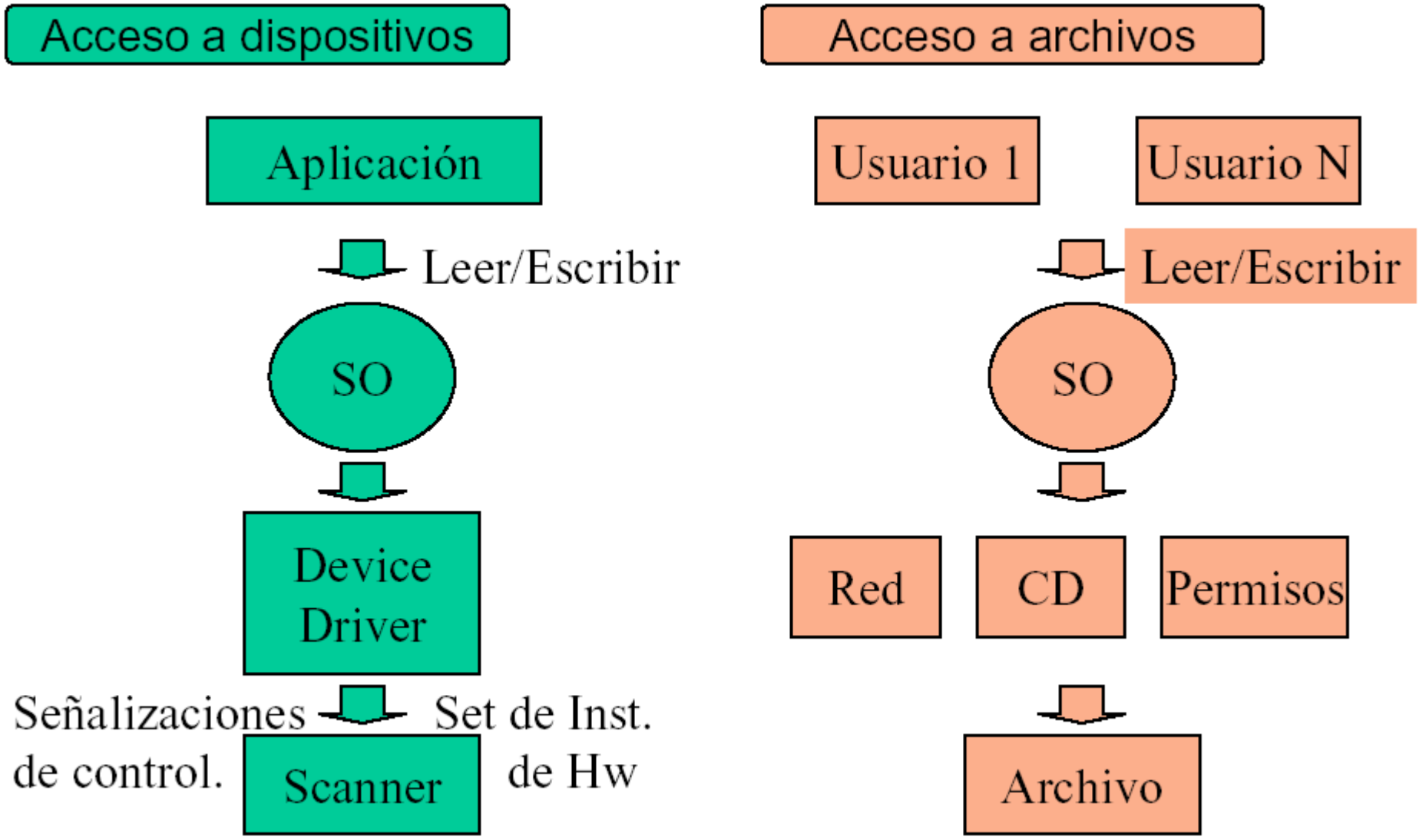
- Utilidades => crear aplicaciones



Ejecución de Programas



Accesos


Acceso a dispositivos
Aplicación
Leer/Escribir
SO
Device Driver
Señalizaciones de control.
Scanner
Set de Inst. de Hw
Acceso a archivos
Usuario 1
Usuario N
Leer/Escribir
SO
Red
CD
Permisos
Archivo

1.1 ¿QUÉ ES UN SISTEMA OPERATIVO?

Primeros Sistemas: Máquina “Desnuda”

Estructura

- Enormes máquinas ejecutadas desde una consola.
- Sistema monousuario.
- El operador era el Programador / Usuario.
- Uso de cinta de papel o tarjetas perforadas.

Primer Software

- Ensambladores (assembler), compiladores.
- Enlazadores (linkers), cargadores (loaders).
- Bibliotecas de subrutinas.
- Manejadores de dispositivos (device drivers).

Hacían uso ineficiente de recursos caros

- Baja utilización de CPU
- Tiempo significativo de inicialización (setup).

¿QUÉ hace un SO?

- El SO es un programa como cualquier otro => ocupa tiempo de CPU
- EL SO dirige al procesador en el uso de los recursos y en el tiempo de ejecución del resto de los programas.
- Parte del SO reside en memoria. Contiene las funciones más importantes
- El resto de la memoria contiene otros programas.
- La distribución de la Memoria también es realizada por el SO, así como los dispositivos de I/O y Procesador.

Objetivos Generales de un SO

Conveniencia

debe facilitar el uso del hardware

Eficiente

debe lograr que los recursos sean utilizados en forma coherente y organizada

Escalable

Un sistema operativo debe ser construido de tal manera que permita

Tipos y Ejemplos..

- Tipos
- Tiempo Real – RTLinux
- Mono-usuario y mono-tarea – Palm Pilot, DOS
- Mono-usuario y multi-tarea – Windows, MacOS
- Multi-usuarios y multi-tarea – Unix, Linux, VMS

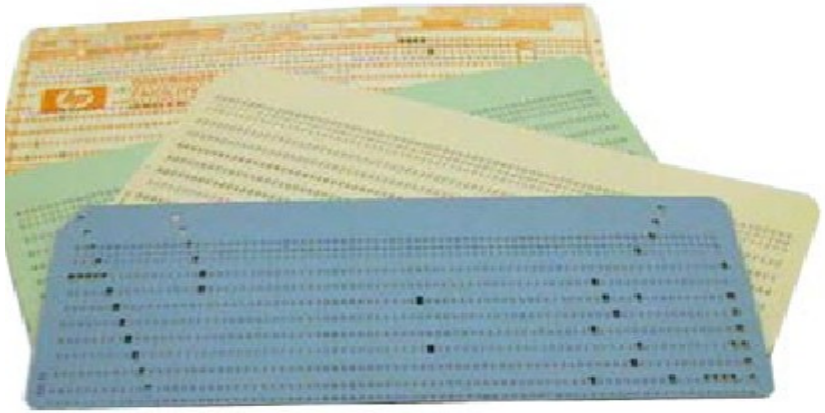
- Ejemplos
- MS-DOS
- UNIX
  - Solaris, HP/UX, DG/UX, SCO, Linux, Irix, AIX
- Windows 95/98/Me/XP
- Windows NT/2000/2003 Server
- CISCO OS 11.2
- 3Com CoreBuilder Extended Switching Software 8.1.1
- MiniDisc Player, Micro-ondas, etc.

Historia

- 1940s y 1950s.
- 1960s.
- 1970s.
- 1980s.
- 1990s.
- 2000 en adelante .

Historia temprana: los 40s y 50s

- 1940s
  - Primeros computadores no incluyen S. O.
- 1950s
  - Se ejecuta un proceso (job) a la vez.
  - Incluyen tecnologías para hacer transiciones “suaves” entre jobs (job-to-job).
  - Sistemas de procesamiento batch simples
  - Los programas y datos son submitidos en cinta de manera consecutiva.

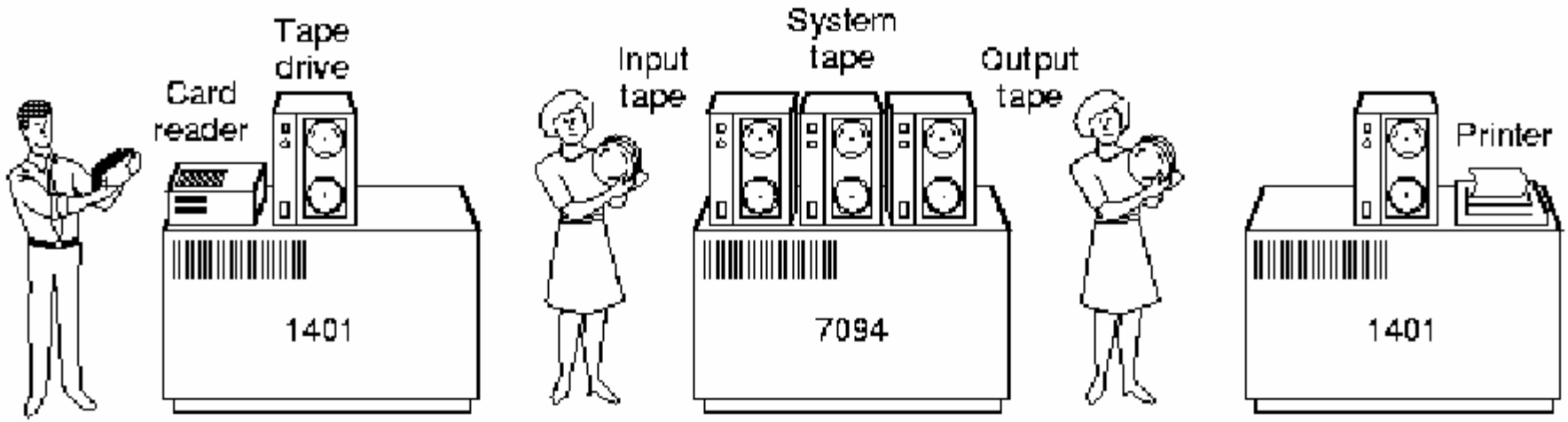


Los 60s

- 1960s
  - Aún hay sistemas de procesamiento batch.
  - Procesan múltiples jobs a la vez (Multiprogramación).
  - Un job puede usar el procesador mientras otros jobs usan los dispositivos periféricos.
  - Sistemas operativos avanzados se desarrollan para dar servicio a múltiples usuarios interactivos.
- 1964
  - IBM anuncia la familia de computadores System/360.

Idea del Proceso

- Proceso: Diseñar el programa, perforar tarjetas, cargar cinta (1401), procesar (7094), imprimir output (1401). (sistema on-line)
- avance: Sistema Batch (juntar varios jobs en una cinta)
- Se imprimen off-line



Los 60s

- Sistemas de Tiempo Compartido (Timesharing)
  - Desarrollados para soportar muchos usuarios interactivos de manera simultánea.
  - Tiempo de turnaround (lapso entre en de envío job y el retorno de sus resultados) es reducido a minutos o segundos.
  - Sistemas de Tiempo Real. Dan respuesta dentro de cierto período de tiempo limitado.

Los 60s

- Advanced Research Projects Agency (ARPA).
  - Departamento de Defensa crea e implementa ARPAnet.
  - Conecta en red los principales computadores de las instituciones fundadoras de ARPA.
  - Capaz de comunicar casi al instante via e-mail.
  - Diseñada para operar sin control centralizado.

Los 70s

- Primeros sistemas timesharing multimodo.
  - Soportan procesamiento batch, timesharing y tiempo real.
  - Computación Personal en estado incipiente.
- Departamento de Defensa desarrolla TCP/IP.
  - Protocolo estándar, establece las reglas y gestiona la comunicación entre aplicaciones sobre ARPANet.
  - Ampliamente usado en ambientes militares y universitarios.
  - Problemas de seguridad. Volumen creciente de información pasa por líneas de comunicaciones vulnerables. Asegura que los mensajes sean enrutados. Detección y corrección de errores.

Los 80s

- Década de computadores personales y estaciones de trabajo (workstations).
- Computación distribuida a los lugares (sites) en los cuales se necesita.
- Computadores personales prueban ser relativamente fáciles de aprender y usar. Interfaces Gráficas de Usuario (GUI).
- Transferencia de información entre computadores via red se hace más económica y práctica.

Los 80s

- Computación Cliente/Servidor se extiende.
  - Clientes requieren varios servicios.
  - Servidores realizan servicios requeridos.
- World Wide Web (WWW)
  - Ubica y gestiona documentos multimediales (páginas)
  - Desarrollo inicial en 1989 en CERN (Tim Berners-Lee).
  - Comparte información via páginas hiper-enlazadas.
  - HTML. HyperText Markup Language (formateo).
  - HTTP. Hypertext Transfer Protocol (transferencia).

Los 90s

Mejora exponencial en rendimiento de hardware

- Potencia de CPU y almacenamiento baratos.
  - Ejecuta programas grandes y complejos en computadores personales.
  - Máquinas económicas para bases de datos extensas.
  - Mainframes poco necesarios.
- Cambio acelerado hacia computación distribuida.
  - Computadores múltiples realizando de manera independiente tareas comunes.

Los 90s

- Soporte de Sistema Operativo para tareas de red se hace estándar.
- Productividad y comunicación aumentan.
- Microsoft se convierte en compañía dominante.
  - Sistema Operativo Windows
  - Emplea muchos conceptos iniciales usados en Sistema Operativo Macintosh.
  - Posibilita a usuarios navegar con facilidad por múltiples aplicaciones concurrentes.

Los 90s

- Tecnología Orientada a Objetos se hace popular en muchas áreas de la computación.
  - Muchas aplicaciones escritas en lenguajes de programación, como C++ o Java.
  - Sistemas operativos orientados a objeto (OOOS). Los objetos representan los componentes del S. O.
  - Conceptos tales como herencia e interfaces.
    - Usados para crear Sistemas Operativos modulares.
    - Más fáciles de mantener y extender que sistemas construidos con tecnologías previas

Los 90s

- Mayoría de software comercial vendido como código objeto.
  - Código fuente no incluido
  - Vendedores esconden información y técnicas de programación propietarios.
- Software gratis y de código abierto se hacen comunes en los 90s.
  - Software de código abierto distribuido con el fuente. Permite a los individuos examinar y modificar el software.
  - Sistema operativo Linux y servidor Web Apache en esta categoría.

Los 90s

- Richard Stallman inicia proyecto GNU.
  - Recrea y extiende herramientas para UNIX AT&T.
  - En desacuerdo con pagar por usar software.
- Open Source Initiative (OSI).
  - Fundada para aumentar beneficios de la programación en código abierto.
  - Facilita mejoras a los productos de software. Permite a cualquiera examinar, depurar y mejorar aplicaciones.
  - Incrementa posibilidad de encontrar y corregir errores menores.
  - Individuos y corporaciones pueden modificar el código fuente, para crear software adaptado que satisfaga necesidades de ciertos ambientes.

Los 90s

- Sistemas Operativos cada vez más amistosos.
  - Características GUI iniciales de Apple ampliamente usadas y mejoradas.
  - Capacidades “plug-and-play” incluidas en los Sistema Operativos, permiten agregar o quitar dinámicamente componentes de

Apple Lisa

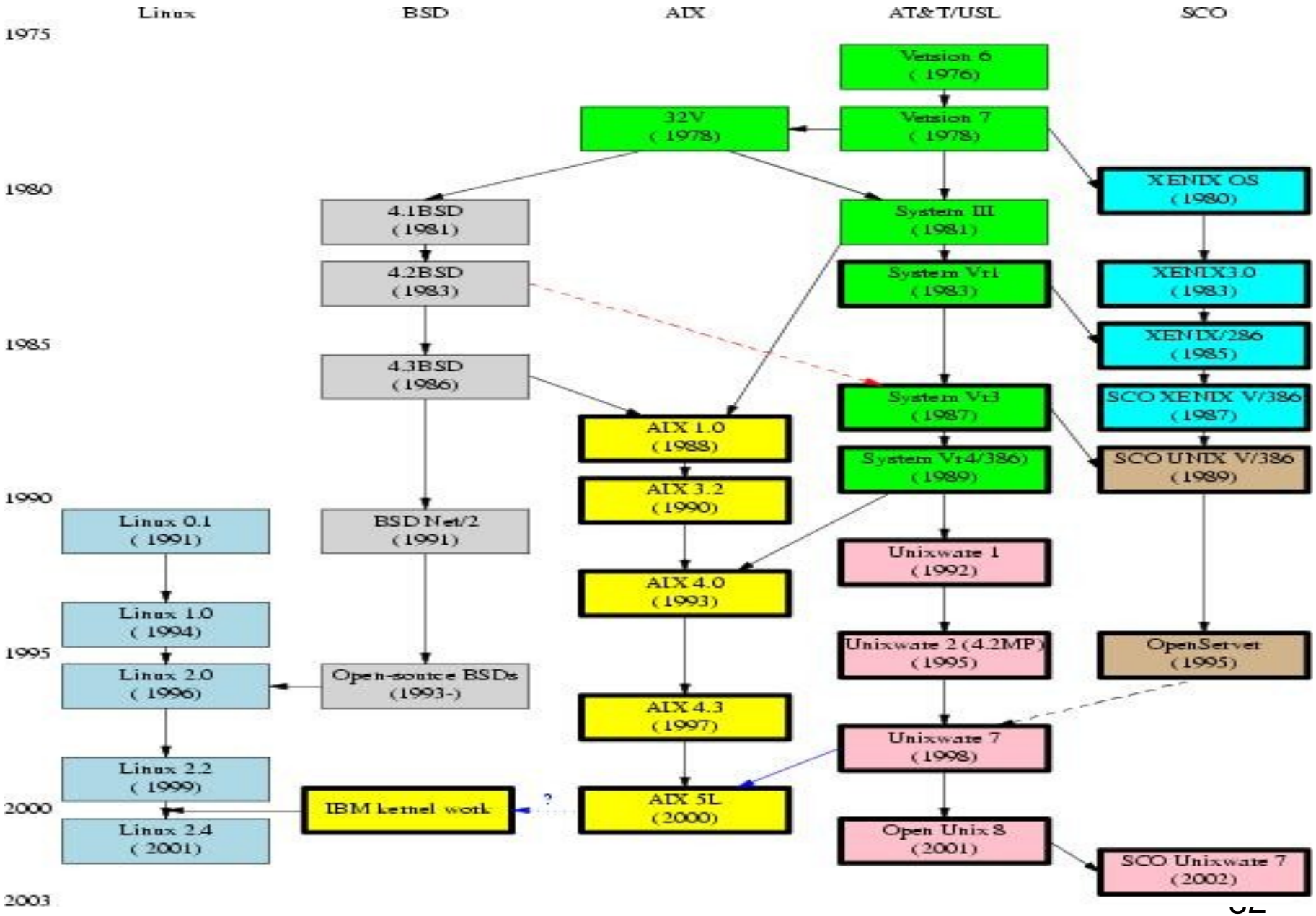
Apple Macintosh

IBM PC

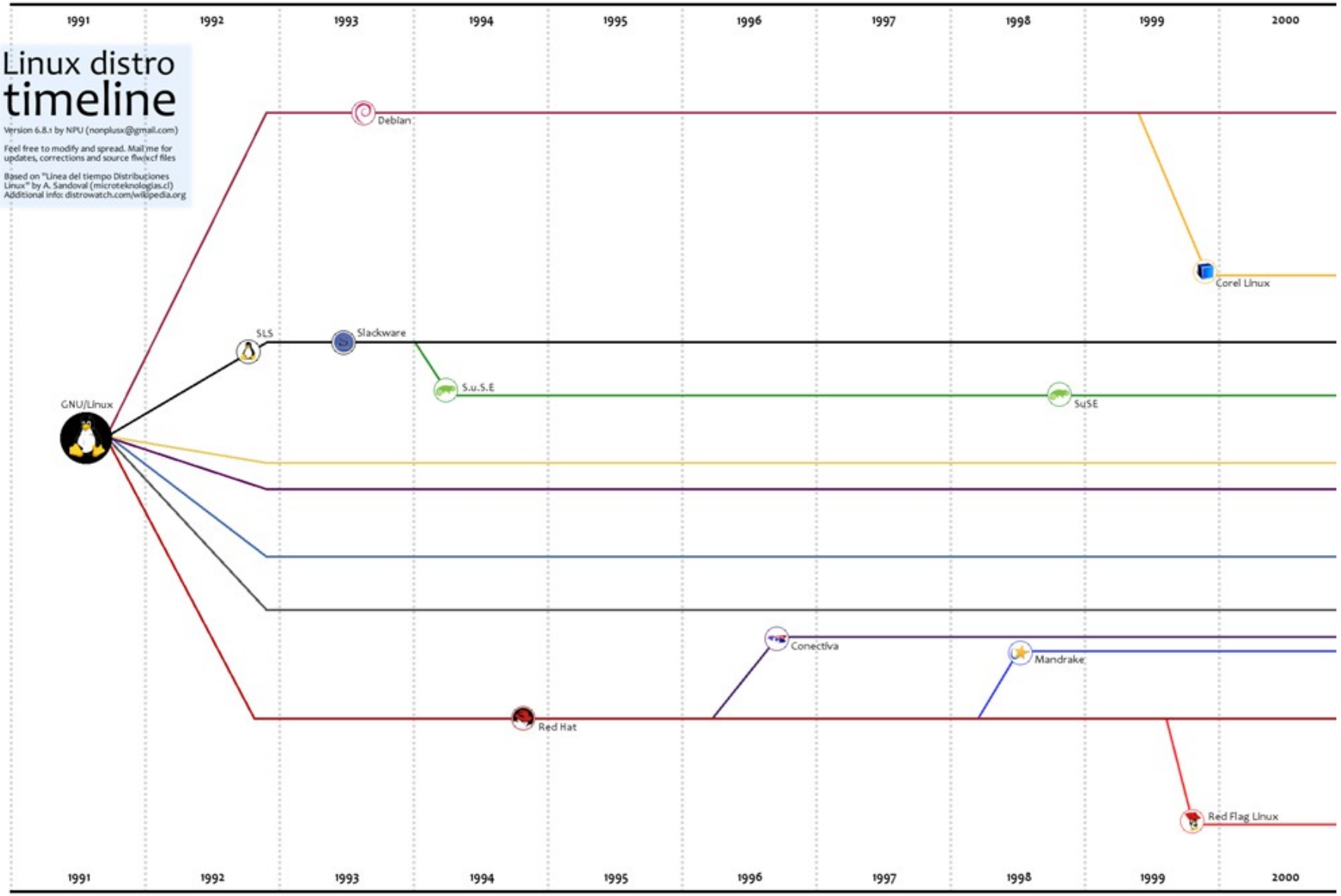
2000 en adelante

- Middleware
  - Enlaza dos aplicaciones separadas, a menudo sobre una red y entre máquinas incompatibles.
  - Particularmente importante para servicios Web. Simplifica la comunicacion entre arquitecturas múltiples.
- Servicios Web
  - Conjunto de estándares relacionados.
  - Piezas de software fáciles de usar en Internet.
  - Posibilita que dos aplicaciones cualquiera se comuniquen e intercambien datos.

Evolución de unix



Evolución de Linux



1.2 SISTEMAS BATCH SIMPLES

- Usuario ≠ Operador.
- Uso extensivo de lectores de tarjetas perforadas.
- Reducción del tiempo de inicialización al agrupar jobs similares (batch).
- Secuenciación automática de jobs – transfiere automáticamente el control de un job a otro.
- Primeros sistemas operativos rudimentarios.
- **Monitor residente**
  - Control inicial en el monitor.
  - El control se transfiere a un job.
  - Cuando el job termina, el control retorna al monitor.

1.2 SISTEMAS BATCH SIMPLES

- Partes del monitor residente
  - Intérprete de tarjetas de control: responsable de leer y ejecutar las instrucciones contenidas en las tarjetas.
  - Cargador (loader): carga los programas del sistema y las aplicaciones en la memoria.
  - Manejador de dispositivo (driver): conoce propiedades y características de un dispositivo de E/S del sistema.
- Problema: bajo rendimiento; la E/S y la CPU no pueden superponerse en su operación; un lector de tarjetas es muy lento.
- Solución: operación fuera de línea (**off-line**); aumenta la velocidad de computación, cargando los jobs en memoria desde cintas o tarjetas e imprime off-line.

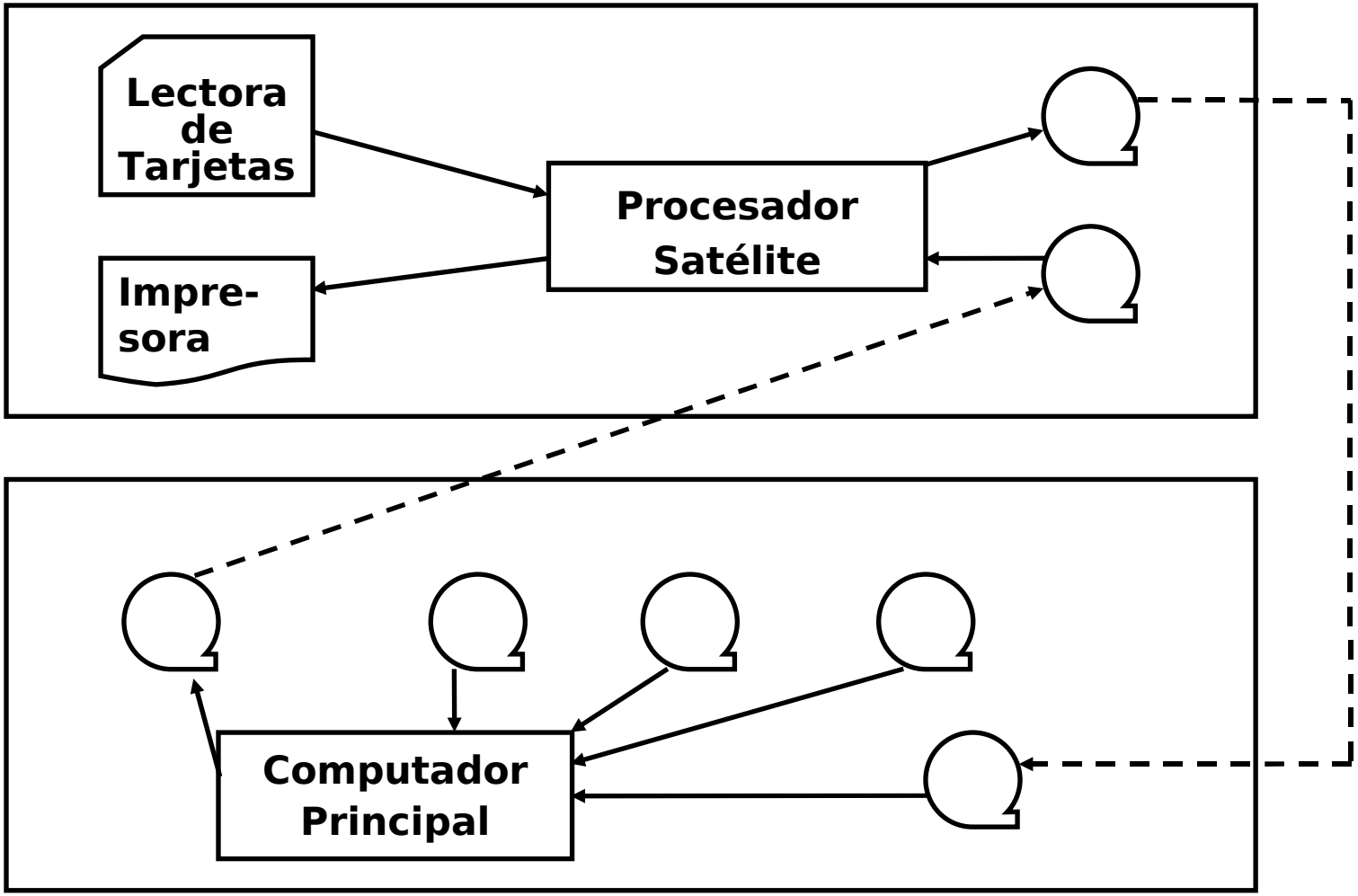
1.2 SISTEMAS BATCH SIMPLES

OPERACIONES EN LÍNEA Y FUERA DE LÍNEA

- **En línea**: los dispositivos periféricos equipados para operación en línea están conectados al procesador, de manera que cualquier requerimiento sea satisfecho de inmediato.
- **Fuera de línea**: manejadas por unidades de control no conectadas al procesador. Esto permite el uso de periféricos, sin sobrecargar directamente el procesador (transferencias entre periféricos lentos).
- **Procesador Satélite**: sistema computacional dedicado para hacer un proceso fuera de línea, por ejemplo, digitación hacia cinta o diskette.

1.2 SISTEMAS BATCH SIMPLES

Sistema Off-line



1.2 SISTEMAS BATCH SIMPLES

Organización de la Memoria en un Sistema Batch Simple

1.2 SISTEMAS BATCH SIMPLES

SPOOLING

- Operación Simultánea de Periféricos en Línea (**S**imultaneous **P**eripheral **O**peration **O**n-**L**ine).
- Es una técnica de Software para:
  - Resolver la gran diferencia de velocidad entre el computador y los dispositivos.
  - Permitir la superposición de E/S de un job con la computación de otro. Mientras se ejecuta un job, el sistema operativo:
    - Lee el siguiente job desde la cola de jobs (job queue).
    - Lee/escribe desde/hacia el dispositivo respectivo.
- El sistema operativo selecciona el siguiente job a ejecutarse, de manera de incrementar la utilización de la CPU.

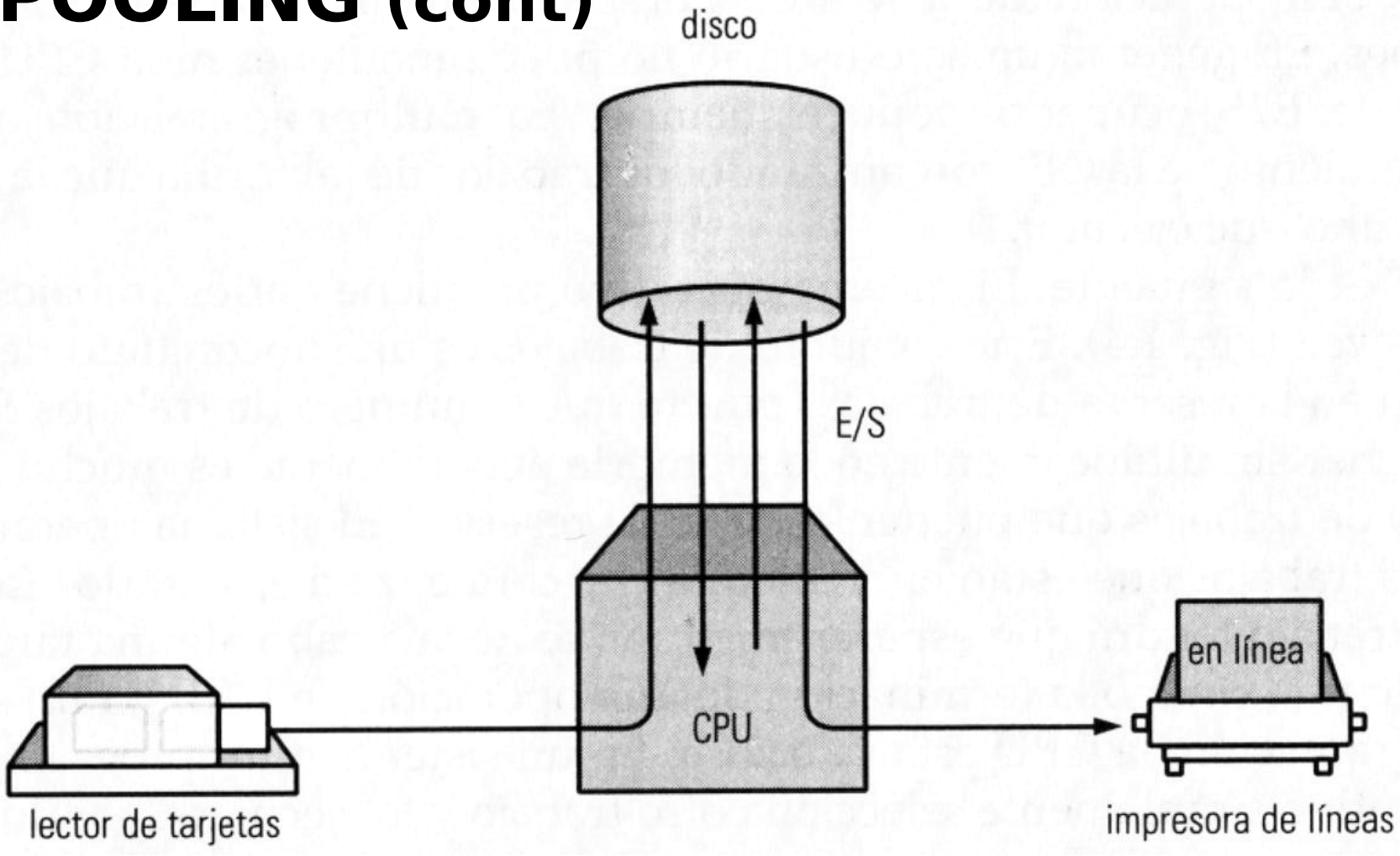
1.2 SISTEMAS BATCH SIMPLES

SPOOLING (cont)

- El Sistema Operativo intercepta la operación, escribiendo un registro a un archivo en disco.
- Ventaja: la velocidad del computador no está restringida por la velocidad de los dispositivos de E/S.
- No se necesita hacer modificaciones a las aplicaciones para cambiar de modo on-line a off-line para la E/S.
- Ganancia real: posibilidad de usar múltiples sistemas de transferencia tarjeta-cinta y cinta-impresora para una CPU.

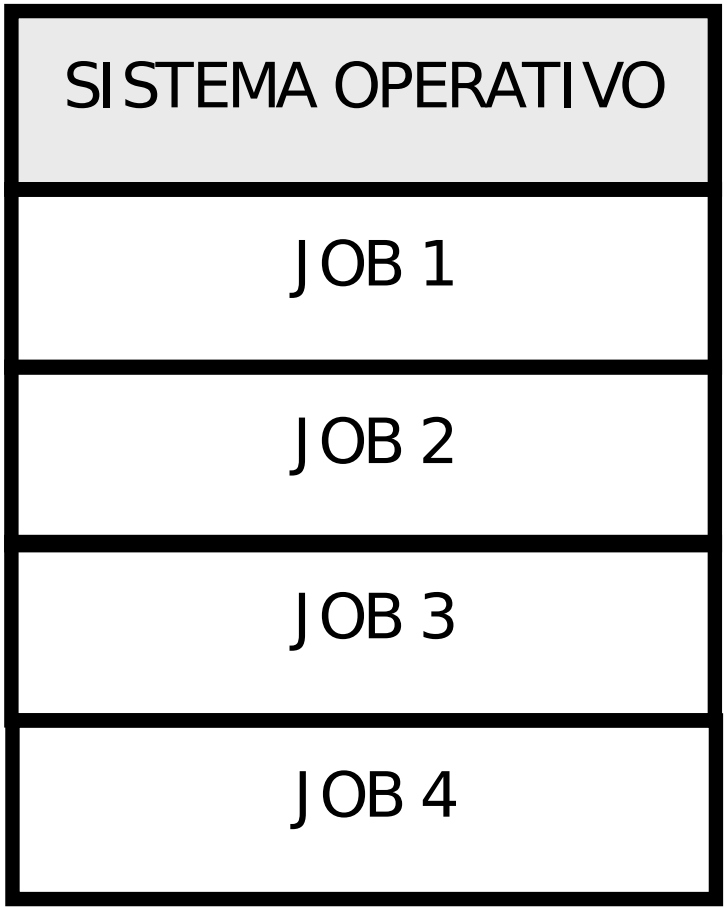
1.2 SISTEMAS BATCH SIMPLES

SPOOLING (cont)



1.3 SIST. BATCH MULTIPROGRAMADO

Hay varios jobs en la memoria principal (job pool) al mismo tiempo y la CPU es multiplexada entre ellos.



1.3 SIST. BATCH MULTIPROGRAMADO

Características del Sistema Operativo necesarias para la Multiprogramación

- Rutinas de E/S proporcionadas por el sistema.
- Administración de memoria: el sistema debe asignar la memoria a varios jobs.
- Itineración de la CPU: el sistema debe elegir entre varios jobs que están listos para ejecutarse.
- Asignación de dispositivos.

1.4 SISTEMAS TIEMPO COMPARTIDO

Time Sharing: Computación Interactiva

- La CPU es multiplexada entre varios jobs que están en memoria y disco. La CPU es asignada a un job sólo si éste está cargado en memoria (Arquitectura Von Neuman).
- Un job es cargado (swapped in) y descargado (swapped out) de la memoria al disco.

1.4 SISTEMAS TIEMPO COMPARTIDO

Time Sharing: Computación Interactiva

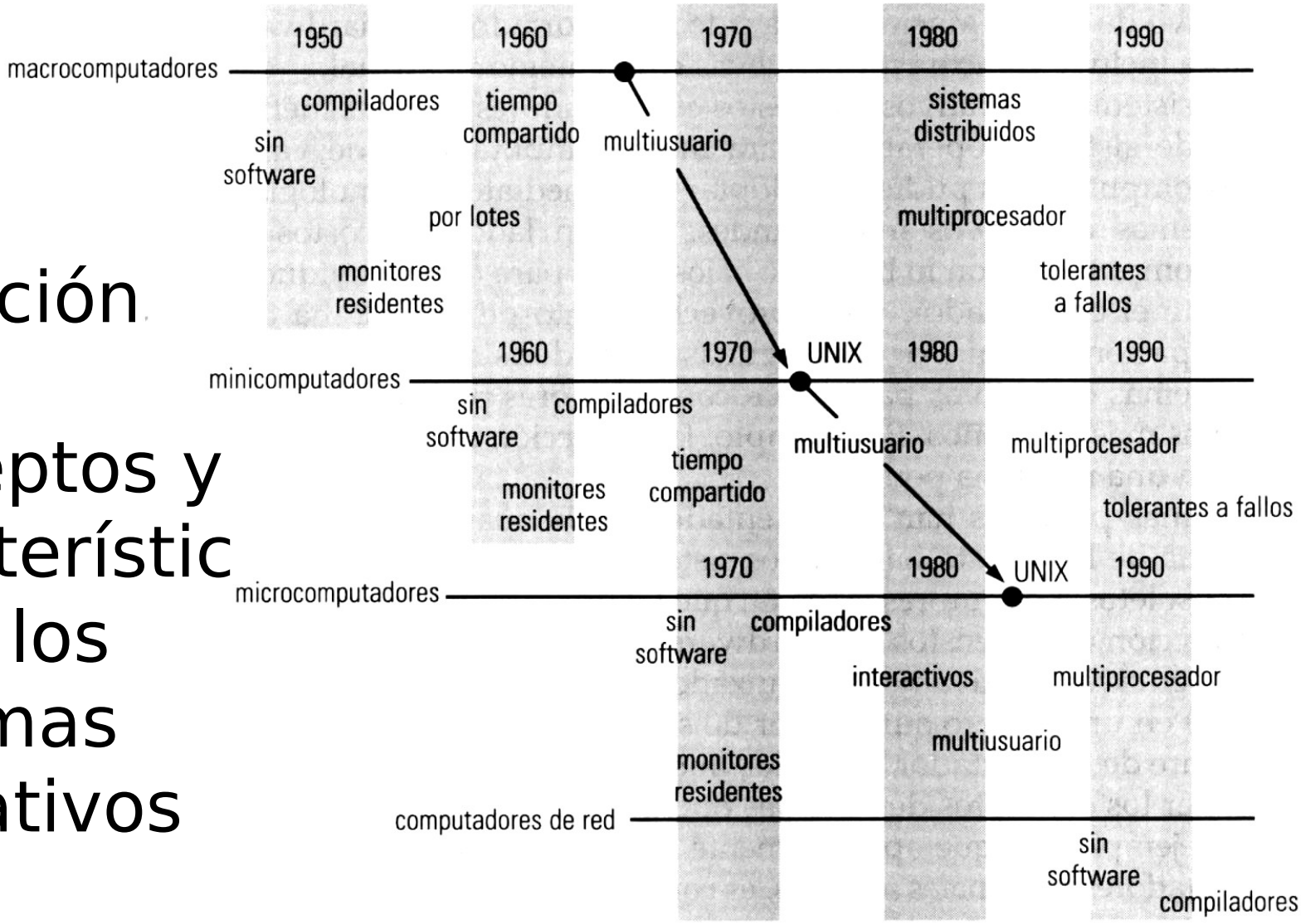
- Hay comunicación en línea (On-Line) entre el usuario y el sistema; cuando el sistema operativo finaliza la ejecución de un comando, busca la siguiente sentencia de control desde el teclado del usuario.
- Debe haber un sistema de Archivos On-Line disponible para que los usuarios puedan acceder a datos y código.

1.5 COMPUTACIÓN PERSONAL

- Un Computador Personal (PC) es un Sistema Computacional dedicado a un solo usuario.
- Proporciona todos los dispositivos de E/S: teclado, mouse, pantalla, impresora, etc.
- El Sistema Operativo está más orientado a la conveniencia del usuario y la interactividad.
- Pueden adoptar tecnología desarrollada para sistemas operativos mayores; a menudo, los individuos hacen un uso básico del computador y no necesitan una utilización avanzada de la CPU o características especiales de protección

1.5 COMPUTACIÓN PERSONAL

Migración de conceptos y características de los Sistemas Operativos



1.6 SISTEMAS PARALELOS

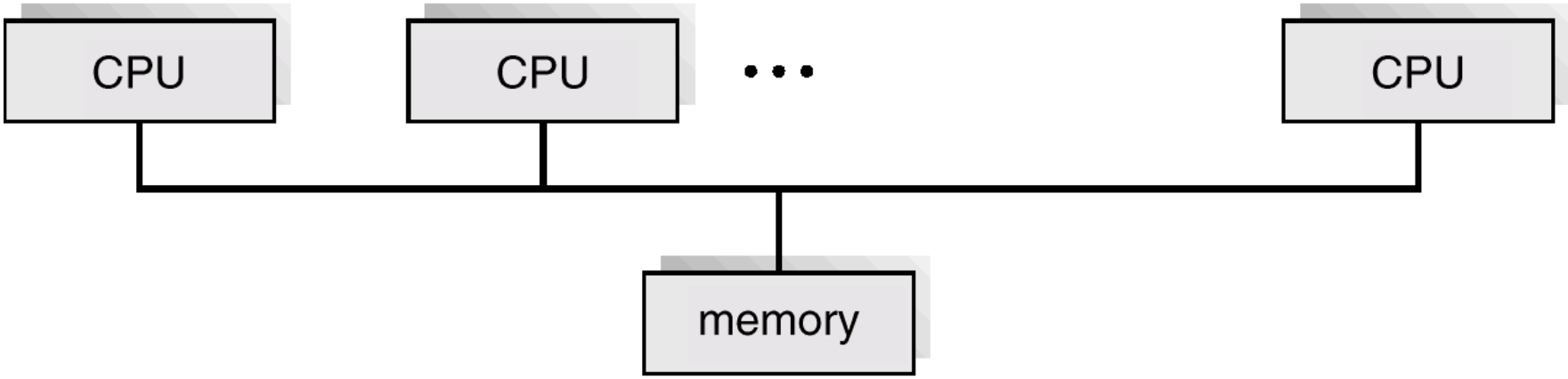
- Los sistemas multiprocesador tienen más de una CPU en estrecha comunicación.
- Son sistemas ***fuertemente acoplados***, donde los procesadores comparten la memoria y el reloj; la comunicación usualmente se lleva a cabo a través de la memoria compartida.
- Ventajas de los sistemas paralelos:
  - Alto rendimiento (throughput)
  - Económicos, comparados con el costo de funcionalidad equivalente en equipos monoprocesador
  - Mayor confiabilidad
    - Poca degradación
    - Sistemas con pocas fallas.

1.6 SISTEMAS PARALELOS

- **Multiprocesamiento Simétrico (SMP)**
  - Cada procesador ejecuta una copia idéntica del sistema operativo.
  - Muchos procesos pueden ser ejecutados a la vez, sin deterioro del rendimiento.
  - La mayoría de los S.O. modernos soportan SMP
- **Multiprocesamiento Asimétrico**
  - A cada procesador se le asigna una tarea específica; los procesadores maestros itineran y asignan el trabajo a los procesadores esclavos.
  - Son muy comunes en sistemas de gran tamaño.

1.6 SISTEMAS PARALELOS

Arquitectura SMP



1.7 SISTEMAS DISTRIBUIDOS

- Distribuyen la computación entre varios procesadores físicamente distantes.
- Son sistemas ***débilmente acoplados***, donde cada procesador tiene su propia memoria y reloj; los procesadores se comunican entre ellos a través de líneas de comunicación.
- Ventajas de los sistemas distribuidos:
  - Compartición de recursos.
  - Aumento de la velocidad de computación, debido a que se comparte la carga de trabajo.
  - Confiabilidad.
  - Comunicación.

1.7 SISTEMAS DISTRIBUIDOS

- Sistema Operativo de Red:
  - proporciona compartición de archivos
  - proporciona esquemas de comunicación
  - se ejecuta independiente de otros computadores en la red
- Sistema Operativo Distribuido:
  - menor autonomía entre computadores
  - da la impresión que hay un solo sistema operativo controlando la red

1.8 SISTEMAS DE TIEMPO REAL

- A menudo, son usados como controladores en aplicaciones dedicadas, como el control de experimentos científicos, sistemas médicos de imágenes, sistemas de control industrial y algunos sistemas de despliegue.
- Tienen restricciones de tiempo muy definidas.

1.8 SISTEMAS DE TIEMPO REAL

- Sistemas de tiempo real “duros” (hard):
  - Almacenamiento secundario muy limitado o está ausente; datos almacenados en memorias de corto plazo o de solo lectura (ROM).
  - Hay conflicto con sistemas time – sharing; no son soportados por S.O. de propósito general.
- Sistemas de tiempo real “blandos” (soft):
  - Utilidad limitada en control industrial o robótica.
  - Más útiles en aplicaciones multimedia o realidad virtual, donde se requiere características más avanzadas del sistema operativo.

1.9 AMBIENTES DE COMPUTACIÓN

- Computación Tradicional
- Computación basada en la Web
- Computación “empotrada” (Embedded)

Conceptos de Hardware y Software

1. Introducción
2. Evolución de los dispositivos de Hardware
3. Componentes de Hardware
4. Soporte de Hardware para Sistemas Operativos
5. Caching y Buffering
6. Visión general del Software
7. Interfaces de Programación de Aplicaciones
8. Compilación, Enlace y Carga
9. Firmware y Máquinas Multinivel
10. Middleware

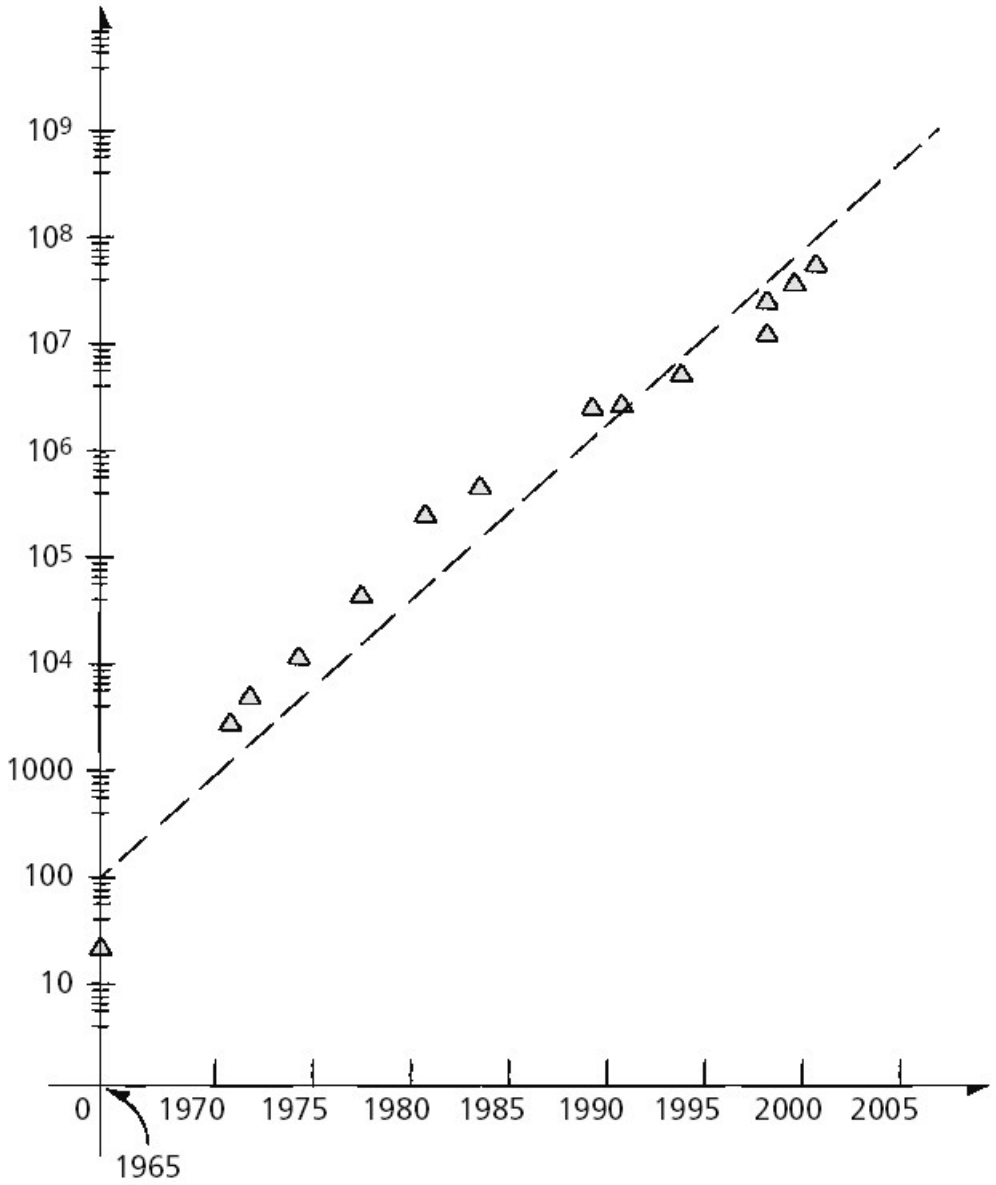
1 Introducción

- Un sistema operativo es mayoritariamente un gestor de recursos.
- Diseño del sistema operativo está atado a los recursos de hardware y software que debe manejar.
    - Procesador(es)
    - Memoria principal
    - Memoria secundaria (discos)
    - Otros dispositivos de E/S
    - Procesos
    - Archivos
    - Bases de datos

2 Evolución de Dispositivos de Hardware

- La mayoría de los sistemas operativos son independientes de la configuración de hardware.
- Los sistemas operativos usan manejadores de dispositivos (device drivers) para llevar a cabo operaciones específicas de dispositivo de E/S.
- Por ejemplo, dispositivos plug-and-play que, cuando son conectados, instruyen al sistema operativo sobre qué driver usar, sin la interacción con el usuario.

2 Evolución de Dispositivos de Hardware



Número de Transistores v/s tiempo (procesador Intel)

al 2009 2300 millones

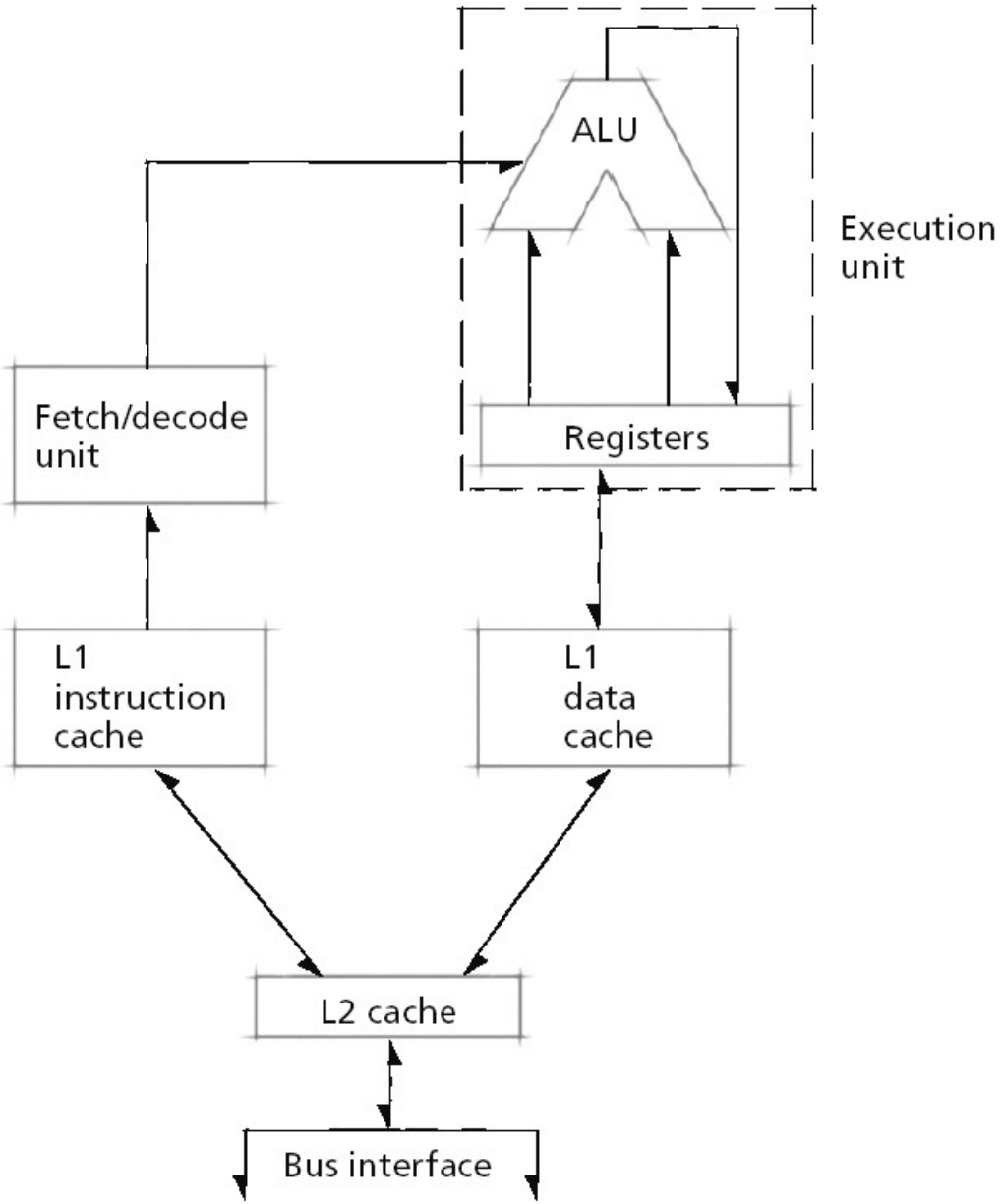
3 Componentes de Hardware

- El hardware del computador consiste de:
  - Procesador(es)
  - Memoria principal
  - Dispositivos de E/S

Procesadores

- Un procesador es hardware que ejecuta lenguaje de máquina:
  - La CPU ejecuta las instrucciones de un programa.
  - El Coprocesador ejecuta instrucciones especiales (ej. gráficos o audio).
  - Los Registros son memorias de alta velocidad ubicadas en el procesador, los datos deben estar en registros antes que el procesador pueda operar sobre ellos.
  - Longitud de Instrucción es el tamaño de una instrucción en lenguaje de máquina, algunos procesadores soportan múltiples longitudes

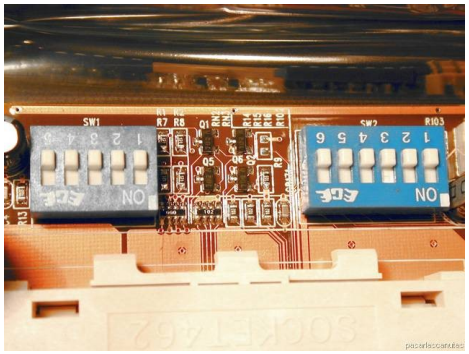
Procesadores



Componentes de un Procesador

Relojes

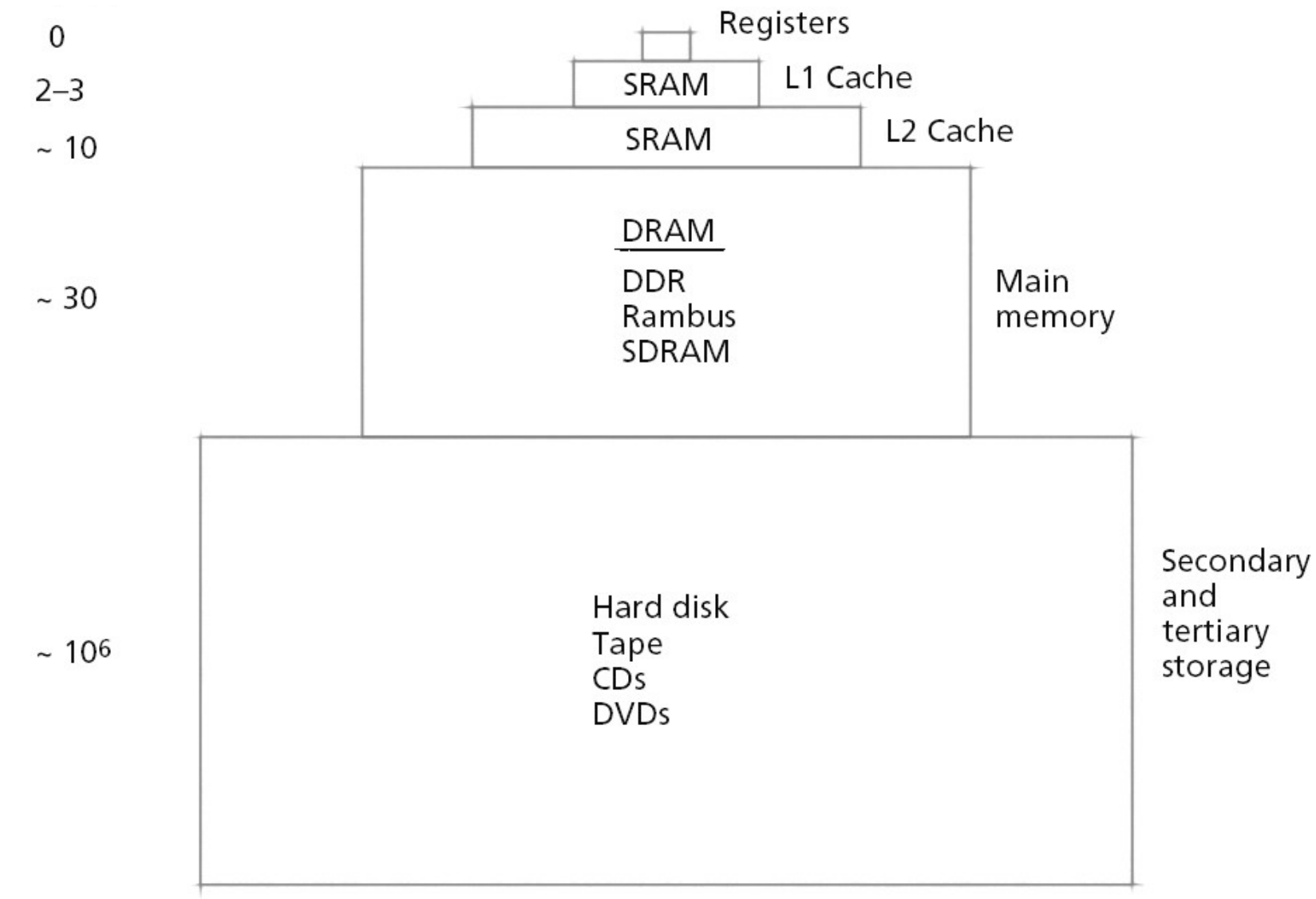
- El tiempo en el Computador se mide en ciclos.
  - Provisto por el generador de reloj del sistema.
  - Las velocidades del Procesador se miden en GHz (x 10^9 ciclos por segundo).



Jerarquía de Memoria

- La jerarquía de memoria es un esquema para categorizar la memoria.
  - La más rápida y más cara arriba, la más lenta y barata abajo.
    - Registros de la CPU
    - Cache nivel 1 (L1)
    - Cache nivel 2 (L2)
    - Memoria Principal
    - Almacenamiento Secundario
    - Almacenamiento Terciario (CDs, DVDs, cintas, diskettes)
  - La Memoria Principal es el dispositivo más lento que puede ser referenciado directamente por el procesador.

Jerarquía de Memoria

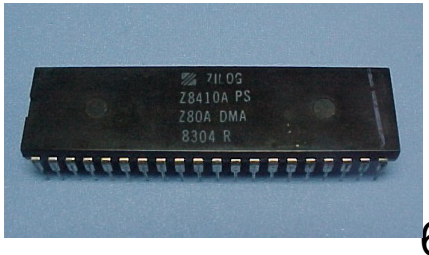


Jerarquía de Memoria

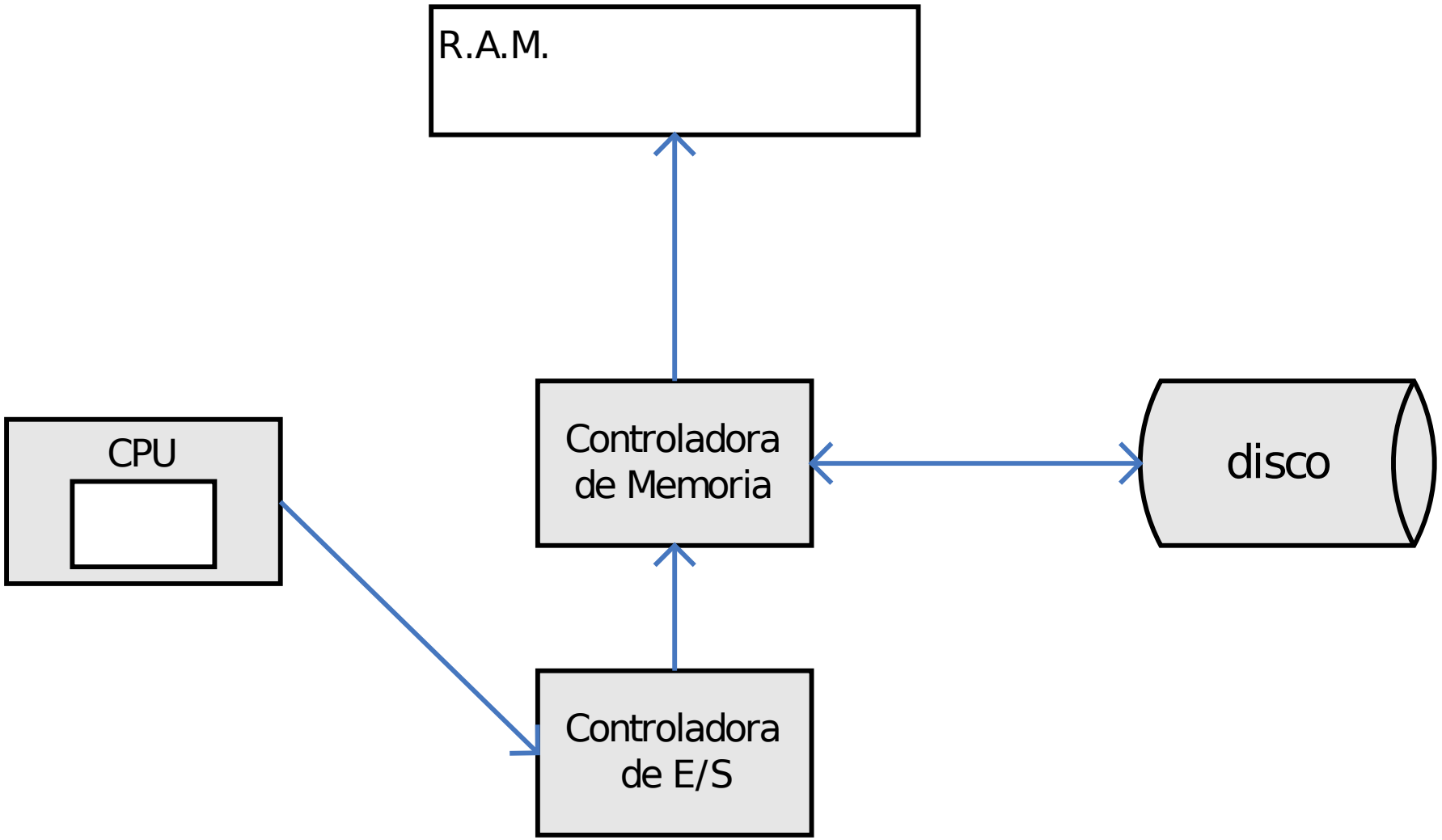
- Memoria Principal.
  - Consiste de memoria volátil de acceso directo (RAM).
  - La CPU puede accesar posiciones de datos en cualquier orden.
- Almacenamiento Secundario.
  - Almacena gran cantidad de datos persistentes, a bajo costo
  - Acceso a datos más lento que en memoria principal, porque el disco tiene componentes electromecánicos.
- Almacenamiento terciario (removible).
  - CDs (CD-R, CD-RW) y DVDs (DVD-R, DVD+R)
  - Diskettes y Zip
  - Cintas

Acceso Directo a Memoria (DMA)

- DMA mejora la transferencia de datos entre la memoria y los dispositivos de E/S.
- Los dispositivos y controladoras transfieren datos directamente hacia y desde memoria principal.
- El procesador queda libre para ejecutar instrucciones de software.
- El canal DMA usa un controlador de E/S para gestionar la transferencia de datos, notificando al procesador cuando se completa la operación de E/S.
- Mejora el rendimiento en sistemas que llevan a cabo gran cantidad de operaciones de E/S (ej., mainframes y servidores).



Acceso Directo a Memoria (DMA)



Robo de Ciclo

- Un punto de conflicto entre los canales y el procesador está en el acceso al almacenam. 1º.
- Como no puede haber más que un solo acceso de cada vez (a un banco de almacenamiento primario), y como es posible que, tanto los canales como el procesador, quieran tener acceso al almacenamiento primario de forma simultánea, en general se le da prioridad a los canales.
- Esto se llama robo de ciclo; los canales, literalmente, roban ciclos al procesador, utilizando un bajo porcentaje de ciclo, consiguiendo una mejor utilización de los dispositivos de E/S.

Canales de E/S

- Por su velocidad, el procesador central está limitado cuando controla operaciones de E/S.
- Un canal es un computador de propósito especial, dedicado al manejo de E/S con independencia del procesador.
- Un canal puede tener acceso directo al almacenamiento principal para almacenar o recuperar información.
- Su utilidad mayor es permitir la actividad concurrente del hardware.

Acceso Directo a Memoria (DMA)

- Se puede mejorar la ejecución en un computador minimizando el número de interrupciones que ocurren durante la ejecución de un programa.
- El Acceso Directo a Memoria requiere de una sola interrupción por cada bloque de caracteres transferidos durante la operación de E/S.
- Esto es significativamente más rápido que el método en el cual el procesador es interrumpido por cada caracter transferido.
- Una vez iniciada una operación de E/S, los caracteres son transferidos al almacenamiento primario en régimen de robo de ciclo.

Acceso Directo a Memoria (DMA)

- Cuando un dispositivo está listo para transmitir un caracter, “interrumpe” al procesador.
- Con DMA, no debe salvarse el estado del procesador, sólo se lo retrasa. Al terminar la transferencia, el procesador resume su operación.
- Es una característica útil en sistemas que soportan un volumen muy grande de transferencia de E/S.
- El Hardware responsable del robo de ciclo y de la operación de los dispositivos se denomina **DMA**.

Dispositivos Periféricos

- Es cualquier dispositivo no requerido por el computador para ejecutar instrucciones de software
- Hay Dispositivos Internos, referidos como dispositivos periféricos integrados:
  - Tarjetas de red, modems y sonido.
  - Disco duro, CD y DVD.
- Dispositivos de caracter transfieren datos un bit a la vez: teclado y mouse.
- Se pueden conectar al computador via puertas y otros buses: Serial, paralela, USB, SCSI, etc.

Dispositivos Periféricos

Almacenamiento secundario y terciario

Permiten el almacenamiento de grandes cantidades de datos fuera de la memoria principal, a un costo menor. Los hay de 2 tipos, **secuencial y directo**:

- **Cinta Magnética**: son básicamente secuenciales. Su costo es ínfimo pero son más restringidas.
- **Disco**: es el dispositivo periférico más significativo, en lo que a sistema operativo se refiere, debido a que es de acceso directo.
- CD: El **disco compacto** es un soporte digital óptico utilizado para almacenar cualquier tipo de información (audio, video, documentos...).

4 Soporte de hardware para Sist. Operat.

- Arquitecturas de Computador contienen:
  - Características del sistema operativo para mejorar el rendimiento del hardware.
  - Características que permiten al sistema operativo forzar la protección.

Procesador

- El procesador implementa mecanismos de protección del sistema operativo.
- Previene a los procesos de accesar instrucciones privilegiadas o memoria.
- Los sistemas computacionales generalmente tienen dos modos de ejecución diferentes:
  - Usuario (o estado problema), donde el usuario sólo puede ejecutar un subconjunto de instrucciones.
  - Monitor o Kernel (o estado supervisor), donde el procesador puede accesar instrucciones y recursos privilegiados.

Procesador

- Protección y gestión de Memoria
  - Previene que los procesos accesen memoria que no les ha sido asignada.
  - Se implementa usando registros del procesador que sólo pueden ser modificados mediante instrucciones privilegiadas.
- Interrupciones y Excepciones
  - La mayoría de los dispositivos envía una señal (interrupción) al procesador cuando ocurre un evento.
  - Una Excepción es una interrupción generada en respuesta a un error.
  - El S.O. puede responder a una interrupción notificando a los procesos que están esperando por tales eventos.

Interrupciones y Escrutinio

- *Escrutinio (polling).* Técnica para permitir que una unidad verifique el **estado** de otra unidad independiente. Si la última no funciona, la primera continúa con lo que estaba haciendo.

- *Interrupción.* Permite que una unidad obtenga la **atención** de otra. La interrupción permite salvar el estado de la unidad interrumpida antes de procesar la interrupción; después, se procesa la interrupción y se restablece el estado de la unidad interrumpida.

Temporizadores y Relojes

- Temporizadores (timers):
  - Un temporizador periódicamente genera una interrupción
  - Los sistemas operativos usan temporizadores para prevenir que algún proceso monopolice el procesador.
- Relojes:
  - Proporcionan una medida de continuidad.
  - Un reloj-hora permite que el S. O. determine la fecha y hora actuales.

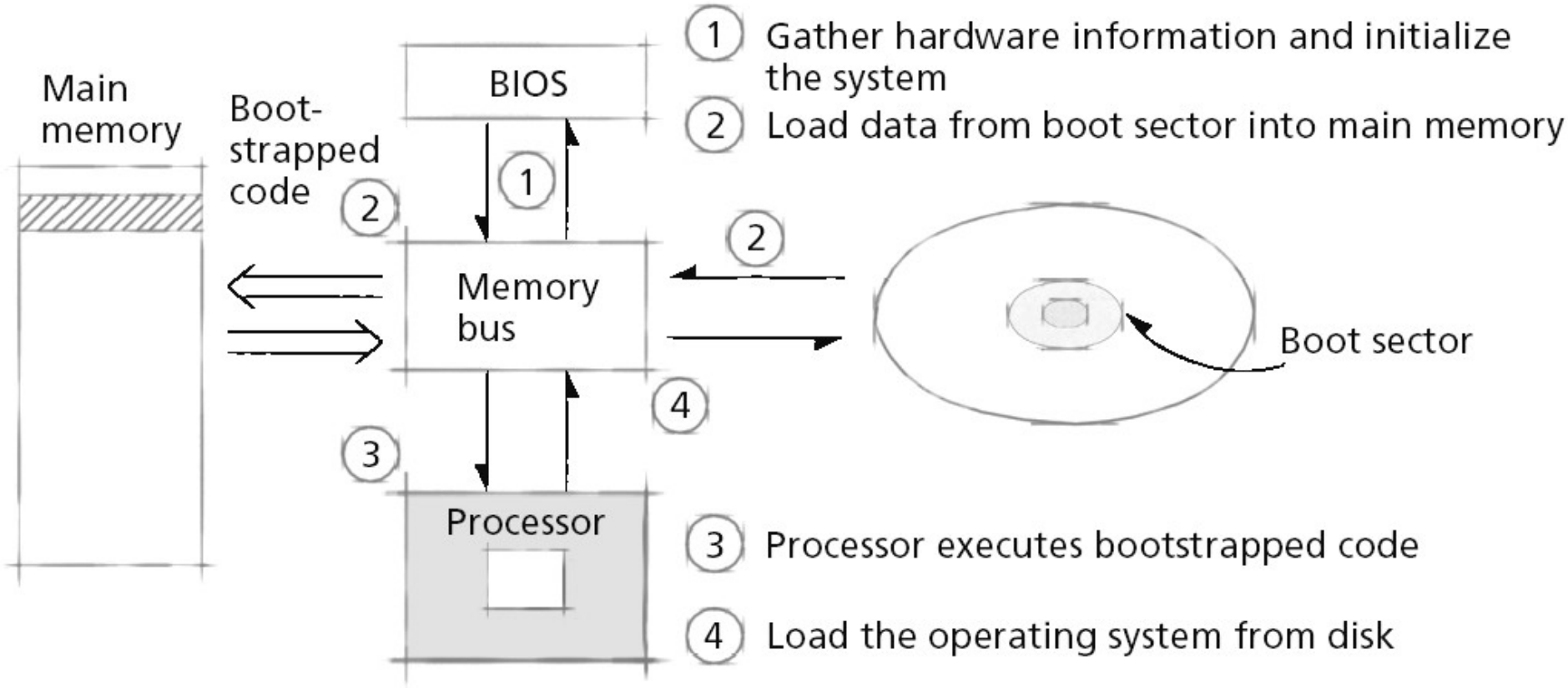
Protección de Almacenamiento

- Es una manera de acotar el espacio de direccionamiento que un programa puede referenciar. Es esencial en los sistemas multiusuario.
- Se puede implementar por medio de “Registros Límite” (dirección superior e inferior del bloque de almacenamiento del programa).
- Las direcciones referidas por el programa son comparadas con el contenido de los registros límite.

Partida (bootstrapping)

- Carga inicial de los componentes del S. O. en memoria.
  - Llevado a cabo por el Sistema Básico de Entrada/ Salida (BIOS). Basic Input/Output System.
  - Inicializa el hardware del sistema.
  - Carga instrucciones en la memoria principal desde el sector de carga (boot sector), ubicado en disco.
  - Si el S. O. no es cargado, el usuario no podrá accesar el hardware del computador.

Partida (bootstrapping)



Plug and Play

- La tecnología Plug and Play permite al sistema operativo configurar nuevamente hardware instalado, sin interacción con el usuario.
- Para soportar plug and play, un dispositivo de hardware debe:
  - Identificarse de manera única ante el sistema operativo
  - Comunicarse con el sistema operativo para indicarle los recursos y servicios que el dispositivo requiere para funcionar apropiadamente.
  - Identificar el driver que soporta el dispositivo y permite al software configurar el dispositivo (ej., asignar el dispositivo al canal DMA)

Registro de Reubicación

- Área de almacenamiento que se encuentra en el procesador, que permite reubicar, en forma dinámica, los programas. El registro de reubicación contiene la dirección base del programa cargado en memoria principal, la cual se suma a cada dirección del programa en ejecución.
- El programador sólo se preocupa que las direcciones referidas por el programa sean relativas a 0, permitiendo independencia del programa del usuario con relación al punto de carga.

5 Caching y Buffering

Cache:

- Memoria relativamente rápida.
- Mantiene copias de los datos que serán accesados pronto.
- Aumenta la velocidad de la ejecución de programas.
- Ejemplos incluyen:
  - Caches de procesador nivel 1 y 2.
  - Memoria Principal puede ser vista como un cache para el disco duro y otros dispositivos de almacenamiento secundario.

5 Caching y Buffering

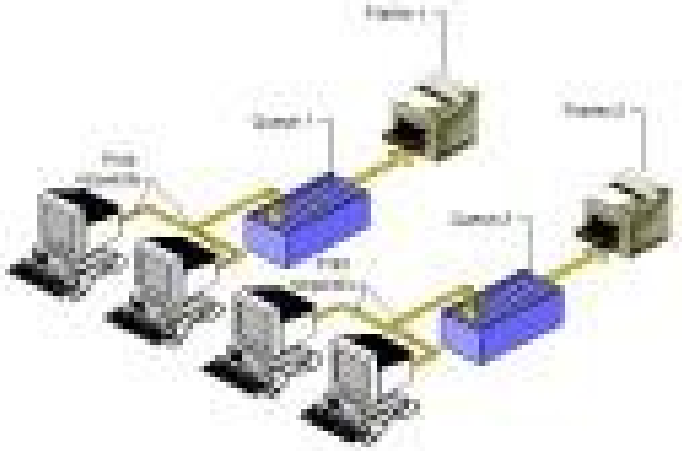
Buffer:

- Área de almacenamiento temporal que mantiene datos durante una transferencia de E/S.
- Es principalmente usado para:
  - Coordinar la comunicación entre dispositivos que operan a diferentes velocidades.
  - Resolver diferencias de tamaño de bloque de transferencia.
  - Almacenar datos para procesamiento asíncrono.
  - Permitir que las señales sean enviadas de manera asincrónica.

5 Caching y Buffering

Spooling:

- Técnica de Buffering en la cual se interpone un dispositivo intermedio, el disco, entre un proceso y un dispositivo de E/S lento.
- Permite que los procesos soliciten operaciones desde un dispositivo periférico sin que sea necesario que el dispositivo esté listo para servir el requerimiento.



6 Visión General del Software

- Lenguajes de Programación:
  - Algunos son directamente entendibles por los computadores, otros requieren traducción.
  - Se clasifican generalmente como:
    - Lenguaje de Máquina.
    - Lenguaje Assembler.
    - Lenguaje de Alto Nivel.

Lenguaje de Máquina y Lenguaje Assembler

- Lenguaje de Máquina
  - Definido en el diseño del hardware del computador.
  - Consiste de conjuntos de números (1s y 0s) que instruyen al computador cómo llevar a cabo operaciones elementales.
  - Un computador puede entender sólo su propio Lenguaje de Máquina.
- Lenguaje Assembler
  - Representa instruccions de Lenguaje de Máquina que utilizan abreviaciones en inglés (símbolos nemotécnicos)
  - El Ensamblador convierte Lenguaje Assembler en Lenguaje de Máquina.
  - Hace más veloz la programación y reduce errores potenciales.

Intérpretes y Compiladores

- Lenguajes de alto nivel.
  - Las instrucciones se parecen al inglés común.
  - Se ejecutan más tareas con menos instrucciones.
  - Requiere de compiladores e intérpretes.
- Compilador.
  - Programa traductor que convierte programas en lenguajes de alto nivel a lenguaje de máquina.
- Intérprete
  - Programa que ejecuta directamente código fuente o código que ha sido reducido a un lenguaje de bajo nivel que no es código de máquina

Intérpretes y Compiladores

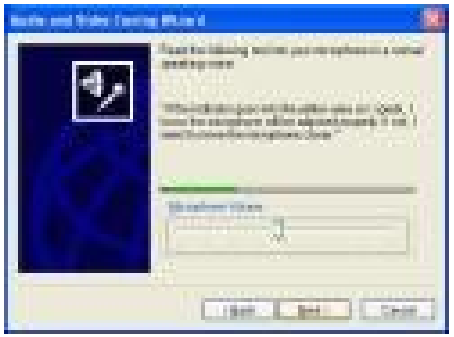
- La ventaja del intérprete es que dado cualquier programa se puede interpretar en cualquier plataforma (sistema operativo).
- En cambio, el archivo generado por el compilador solo funciona en la plataforma en donde se lo ha creado.
- Hablando de la velocidad de ejecución, un archivo compilado es de 10 a 20 veces más rápido que un archivo interpretado.

Lenguajes de alto nivel

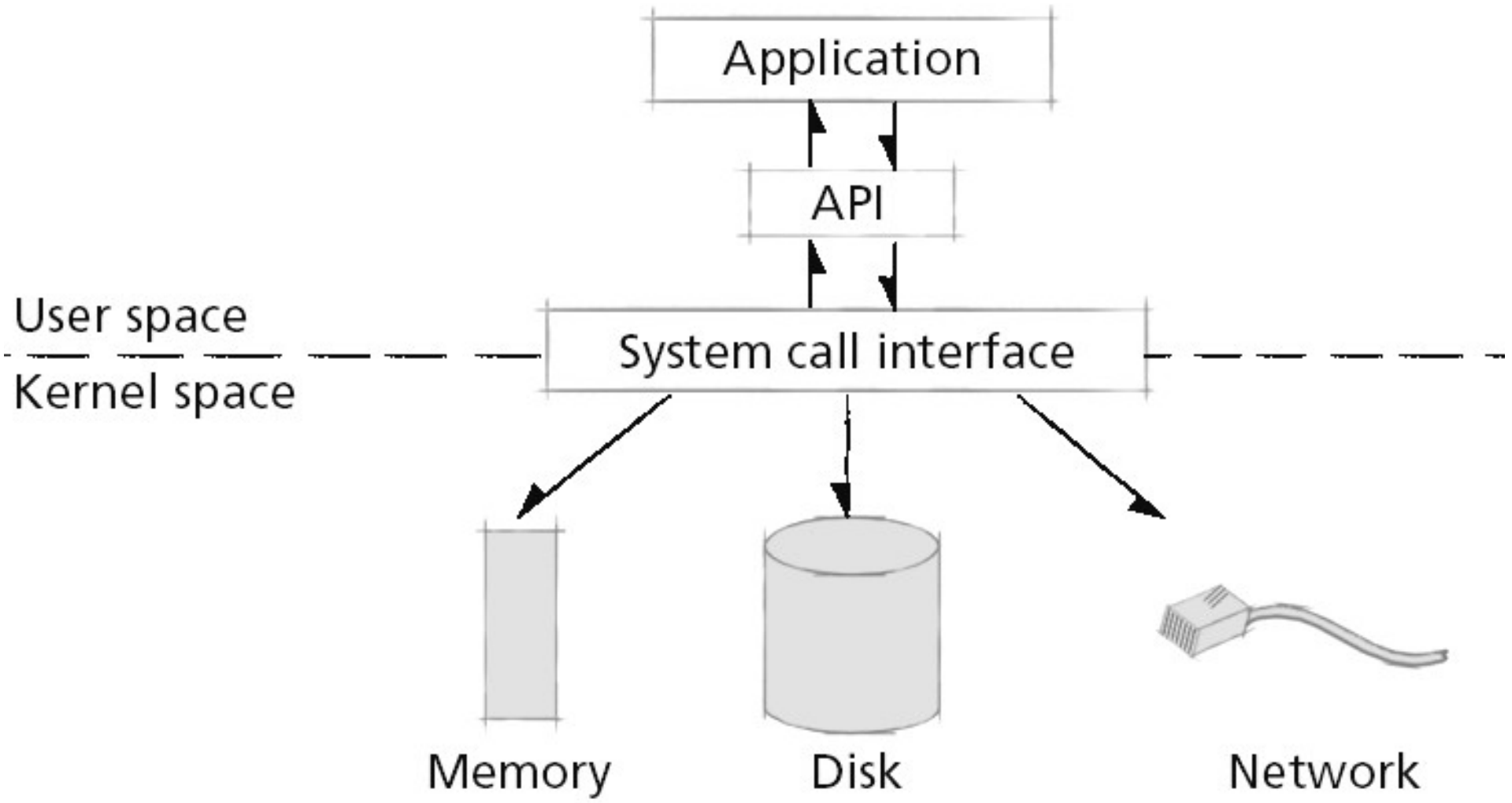
- Lenguajes de alto nivel populares
  - Típicamente procedurales u orientados a objeto (OO)
  - Fortran: aplicaciones científicas e ingenieriles.
  - COBOL: aplicaciones de negocio que manipulan grandes volúmenes de datos.
  - C: Lenguaje de desarrollo del S. O. UNIX.
  - C++/Java: Lenguajes populares OO.
  - C#: Lenguaje de desarrollo OO para plataformas .NET.

7 Interfaz de Programación de Aplicaciones

- Conjunto de rutinas.
- Programadores usan rutinas para solicitar servicios al sistema operativo.
- Programas llaman funciones API, las que pueden accesar el S. O. mediante llamadas al sistema.
- Ejemplos de APIs:
  - Estándar de Interfaz Portable de Sistema Operativo (POSIX).
  - API de Windows.



7 Interfaz de Programación de Aplicaciones

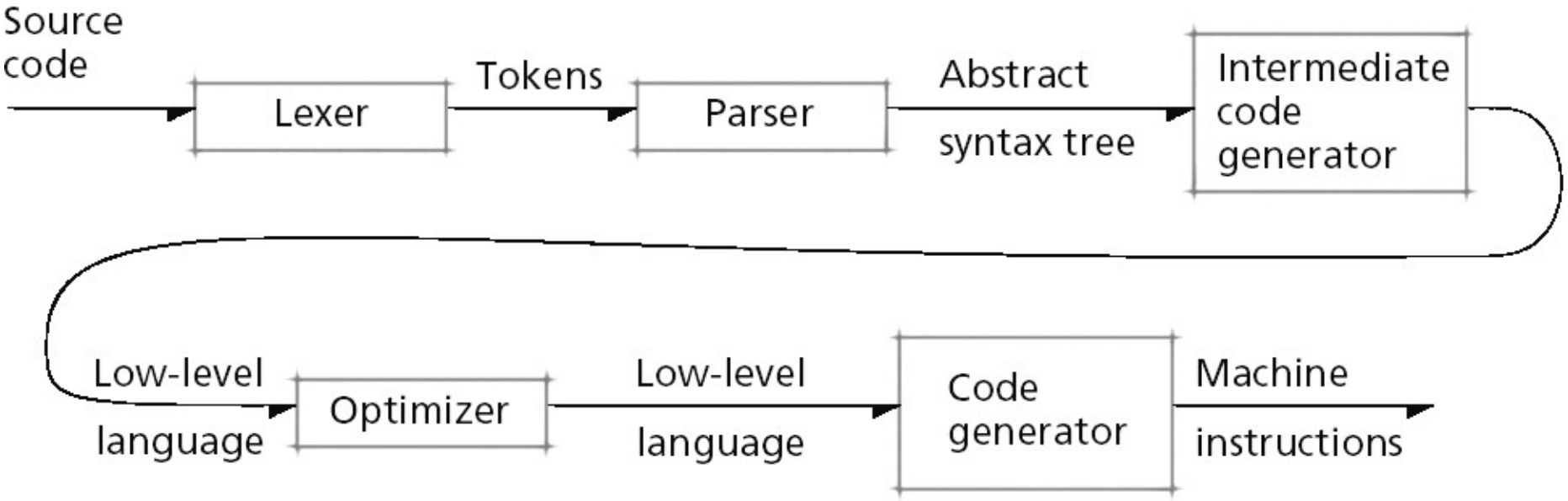


8 Compilación, Enlace y Carga

- Antes que un programa de alto nivel pueda ejecutarse, éste debe ser:
  - Traducido a Lenguaje de Máquina.
  - Enlazado con varios otros programas en Lenguaje de Máquina de los cuales él depende.
  - Cargado en memoria.

Compilación

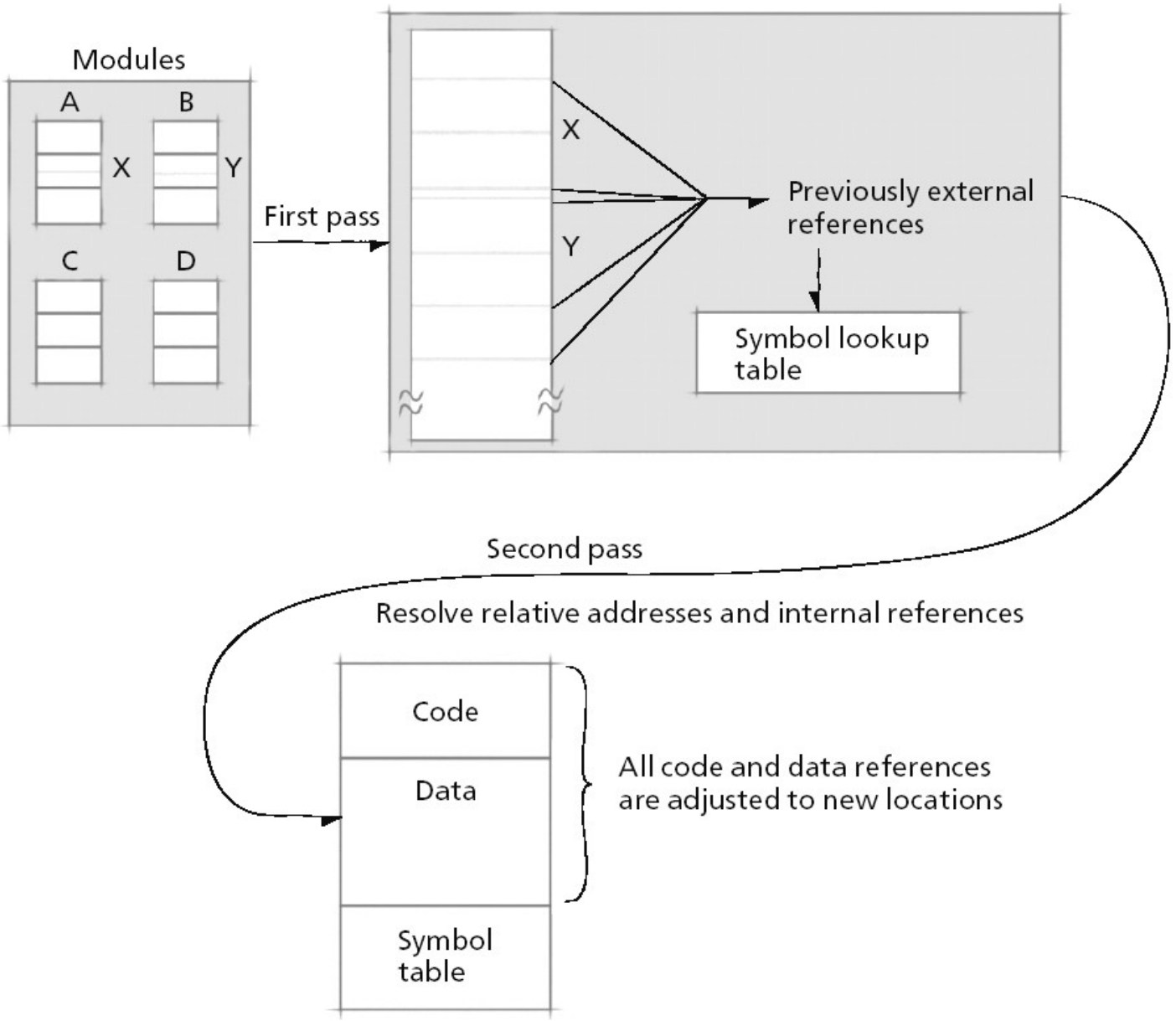
- Traducción de código de alto nivel a código de máquina
- Acepta código fuente como entrada y retorna código objeto.



Enlace

- Crea una sola unidad ejecutable.
- Integra módulos precompilados denominados bibliotecas (library) referenciados por un programa.
- Asigna direcciones relativas a las diferentes unidades de programas o datos.
- Resuelve todas las referencias externas entre subprogramas.
- Produce un módulo integrado llamado módulo de carga.
- Se puede realizar en tiempo de compilación, antes de la carga, en tiempo de carga o de ejecución.

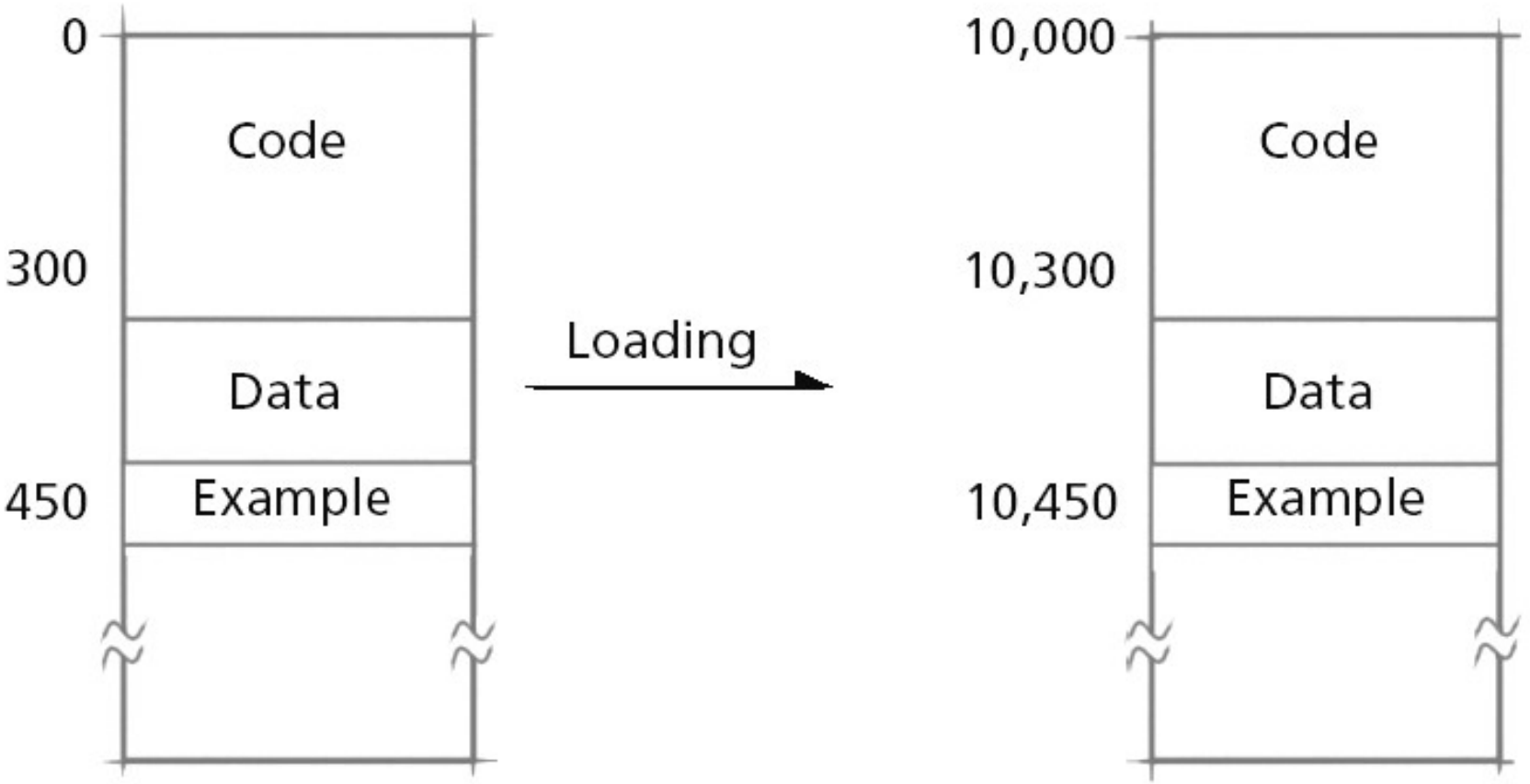
Enlace


Modules
A
B
X
Y
C
D
First pass
X
Y
Previously external references
Symbol lookup table
Second pass
Resolve relative addresses and internal references
Code
Data
Symbol table
All code and data references are adjusted to new locations

Carga

- Un Cargador:
  - Convierte direcciones relativas a físicas.
  - Ubica cada instrucción y dato en memoria principal.
- Técnicas de carga:
  - Absoluta. Ubica un programa en la dirección especificada por el programador/compilador.
  - Reubicable. Reubica las direcciones del programa para que correspondan con su posición real en la memoria.
  - Dinámica. Carga módulos de un programa donde sea necesario para su uso.

Carga


0
Code
300
Data
450
Example
Loading
10,000
Code
10,300
Data
10,450
Example

8 Compilación, Enlace y Carga

- Cargador Absoluto: carga en direcciones específicas indicadas en el programa en lenguaje de máquina.
- Cargador de Reubicación: carga un programa en varios puntos, dentro del almacenamiento 1º, dependiendo de la disponibilidad de almacenamiento, al momento de realizar la carga.
- No siempre la carga la realiza el Loader, pero esa es la tendencia. Puede ser el compilador, el usuario, etc.

Cargadores y Editores de Enlace

- Un Cargador de Enlace, en el momento de carga, combina los programas requeridos y los carga directamente en el almacenamiento primario.
- Un Editor de Enlace ejecuta también la combinación de programas requeridos, pero crea una ”imagen de carga” (Core Image) que es guardada en almacenamiento secundario como futura referencia.
- El editor puede guardar la imagen de carga completa.

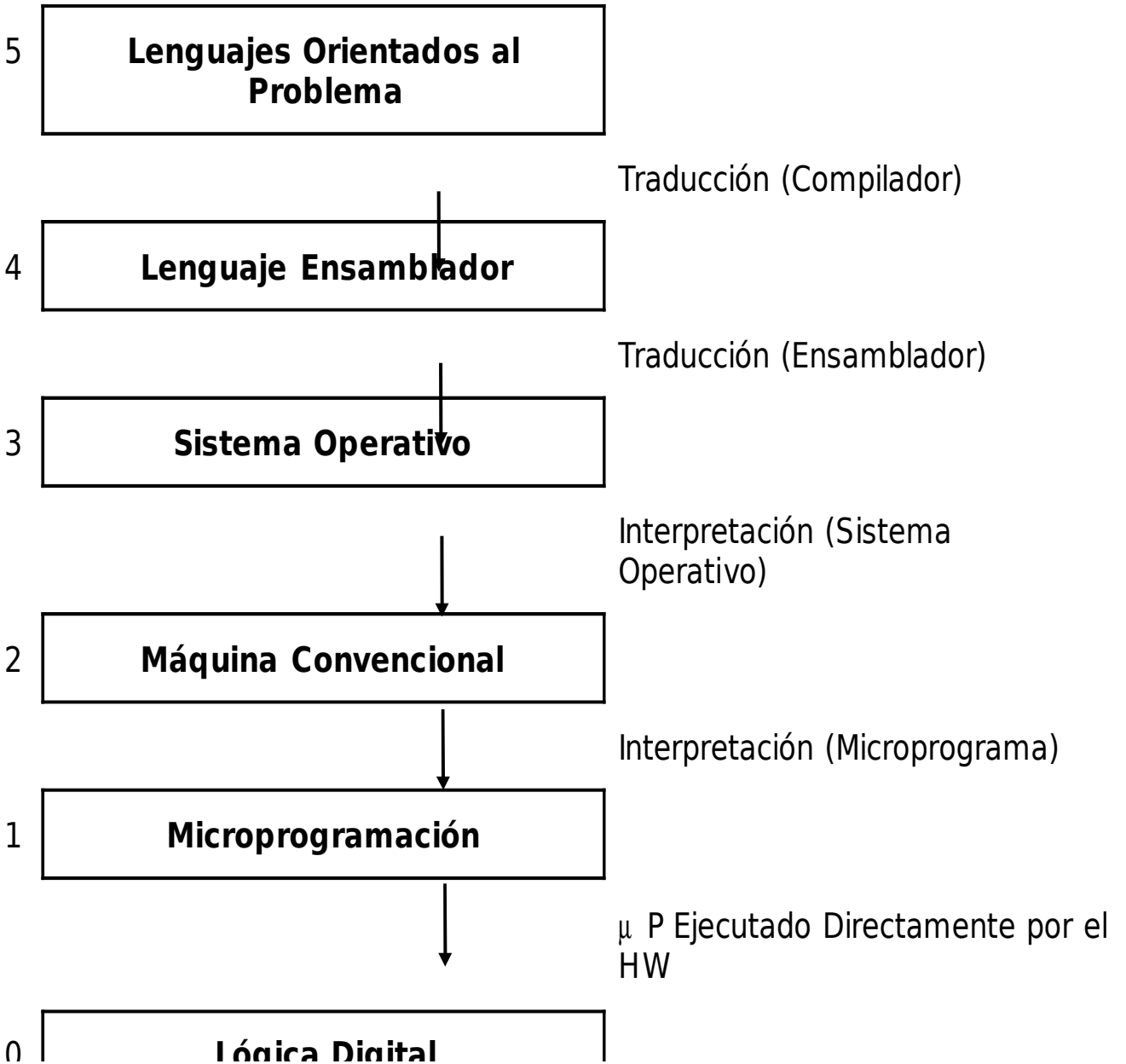
Código Reentrante

- Todo programa consta de instrucciones y datos.
- Cuando el programa se ejecuta, hay una parte fija (instrucciones y constantes) y otra modificable (variables). Al ejecutar un programa por varios usuarios en forma simultánea, cargar el programa completo para cada usuario ocuparía demasiada memoria.
- Si se divide el programa en parte fija y variable, la fija se carga una sola vez y, para cada usuario, se carga una copia de la parte variable.
- La parte invariable es llamada Código

9 Firmware y Máquinas Multinivel

- Un Computador digital es una máquina que puede resolver problemas ejecutando ciertas instrucciones.
- Se llama Programa a la secuencia de instrucciones que dice cómo ejecutar cierta tarea. Los circuitos pueden reconocer y ejecutar directamente un conjunto limitado de instrucciones simples, cuyo conjunto total forma el **Lenguaje de Máquina**.
- Debido a lo elemental de este lenguaje, es muy tediosa su utilización, siendo conveniente agrupar las instrucciones, formando un nivel superior.

9 Firmware y Máquinas Multinivel



9 Firmware y Máquinas Multinivel

- Hay 2 métodos para ejecutar un programa escrito en lenguaje de nivel superior.
- El primero, sustituye cada instrucción del **programa fuente** por una secuencia equivalente de instrucciones en lenguaje de nivel inferior.
- El resultado es un nuevo programa (**objeto**), escrito totalmente con instrucciones del nivel inferior, el cual será ejecutado por el computador.
- Esta técnica se denomina **Traducción**.
- En esta categoría están los **Compiladores** (para los lenguajes de alto nivel) y los **Ensambladores** (para lenguaje

9 Firmware y Máquinas Multinivel

Además, existen los Macroprocesadores:

- Se desea aumentar la velocidad de proceso y reducir errores de codificación.
- Para acelerar el proceso de codificación de un programa assembler, se desarrollan los Macroprocesadores, incorporados en los ensambladores.
- Una Macroinstrucción está hecha sobre la base de una serie de instrucciones en lenguaje ensamblador.
- La Macroexpansión genera en una serie de instrucciones correspondientes a la macroinstrucción.

9 Firmware y Máquinas Multinivel

- Otra técnica es escribir un programa en lenguaje de nivel inferior que lea programas escritos en nivel superior, examine cada instrucción y ejecute directamente la secuencia equivalente de instrucciones del nivel inferior.
- Esta técnica se llama **Interpretación** y el programa que la lleva a cabo, Intérprete, el cual no produce programa objeto.
- La ejecución es más lenta que en los lenguajes compilados, puesto que deben traducir las instrucciones cada vez que éstas se ejecutan.

9 Firmware y Máquinas Multinivel

- Esta agregación de niveles puede continuar, donde cada lenguaje usa su predecesor como base.
- La mayoría de los computadores modernos consta de más de 2 niveles.

9 Firmware y Máquinas Multinivel

Nivel 0

- Es el Hardware mismo. Sus circuitos ejecutan directamente los programas escritos en el nivel 1.
- A este nivel se le denomina **Nivel de Lógica Digital**, y está constituido por las compuertas lógicas y conexiones entre ellas.

9 Firmware y Máquinas Multinivel

Nivel 1

- Conforma un verdadero nivel de lenguaje de máquina; a diferencia del nivel 0, aquí existe el concepto de programa como conjunto de instrucciones a ejecutar.
- A este programa se le llama Microprograma y debe interpretar las instrucciones del nivel 2.
- Se denomina **Nivel de Microprogramación**.
- Aquí aparece el concepto de memoria fija:
  - Programas en microcódigo ejecutados desde un almacenamiento de control de alta velocidad.
  - Programas de uso común dentro de memorias ROM y PROM.

9 Firmware y Máquinas Multinivel

Nivel 1 (cont)

- Una instrucción de máquina puede estar conformada por varias microinstrucciones (un microprograma).
- La microprogramación introduce programación por debajo del lenguaje de máquina convencional, definiendo las instrucciones de éste, mecanismo típico de las arquitecturas modernas.
- Los microprogramas se procesan en el almacenamiento de control.

9 Firmware y Máquinas Multinivel

Nivel 1 (cont)

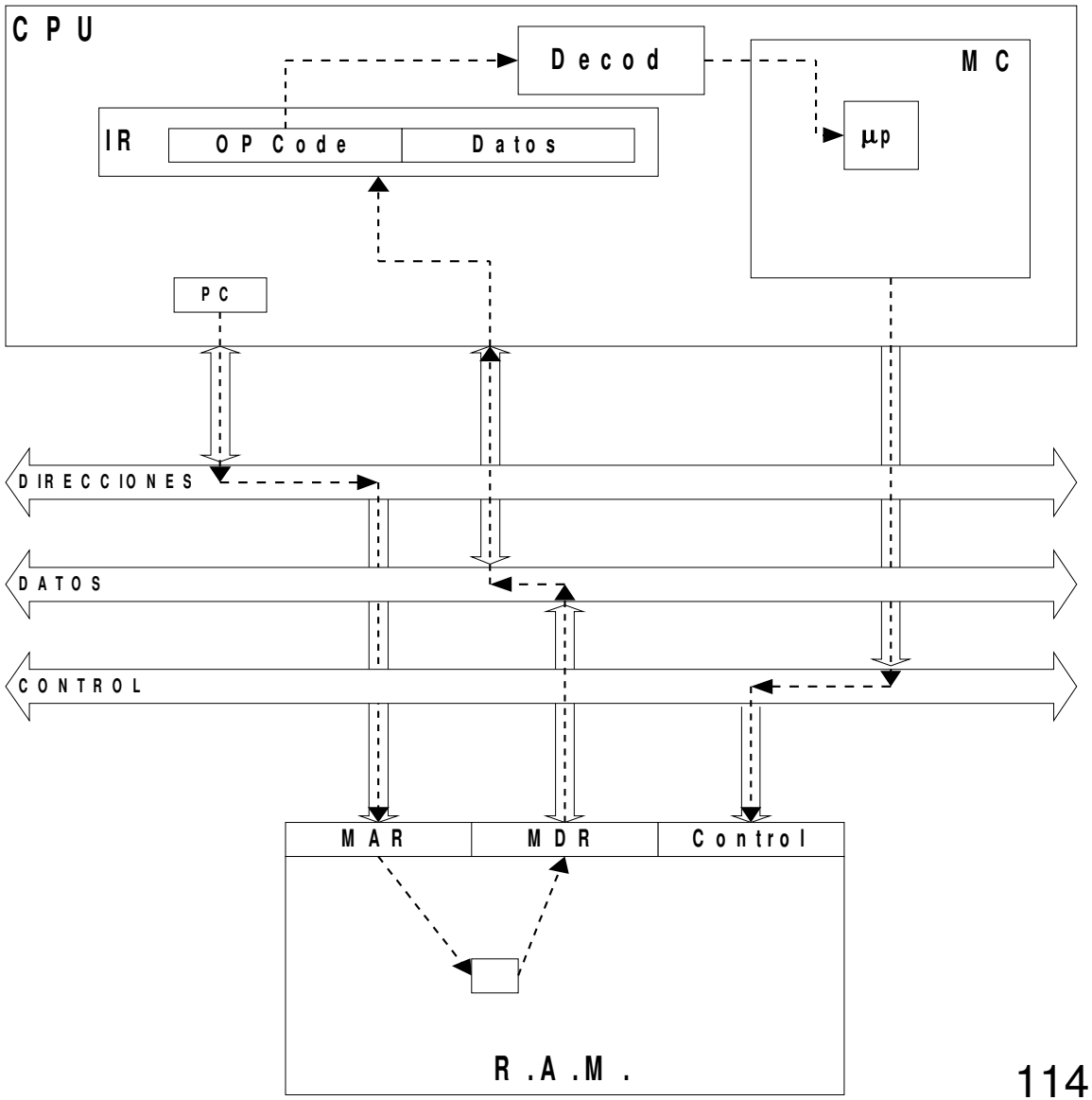
- La decisión de qué funciones implementar en microcódigo se fundamenta en mejorar el rendimiento de la ejecución de un sistema.
- Colocar secuencias de instrucciones de uso más frecuente en el **Firmware** en lugar del Software disminuye el tiempo de ejecución.
- Funciones del Microcódigo:
  - Manejo de interrupciones.
  - Cambio de contexto.
  - Secuencia de llamadas y retorno del procedimiento.
  - Emulación.

9 Firmware y Máquinas Multinivel

Nivel 1 (cont)

MEMORIA FIJA

Ejecución de una instrucción



9 Firmware y Máquinas Multinivel

Nivel 2

- Este es el nivel de **Máquina Convencional**, porque era, por tradición, el lenguaje de máquina que se ejecutaba en el hardware, hasta la aparición del microcódigo. Una instrucción en este lenguaje equivale a un microprograma.
- Aquí, las instrucciones referencian registros específicos del computador y procesan los datos en una forma física determinada.
- Los primeros sistemas computacionales se programaban directamente en lenguaje de máquina convencional.

9 Firmware y Máquinas Multinivel

Nivel 3

- Normalmente es un nivel híbrido. La mayoría de las instrucciones están también en el nivel 2, más un nuevo conjunto de instrucciones.
- Estas últimas son ejecutadas por un intérprete que actúa en nivel 2, al que se llama **Sistema Operativo**.
- Las instrucciones del nivel 3 idénticas a las del nivel 2 son ejecutadas directamente por el microprograma del nivel 1, por esto se le llama híbrido.

9 Firmware y Máquinas Multinivel

Nivel 4

- Existe una gran diferencia entre los niveles 3 y 4.
- Los inferiores están concebidos para ser usados en Programación de Sistemas, mientras que el 4 y superiores son para Aplicaciones.
- Otra diferencia es la naturaleza del lenguaje: para los niveles 1 al 3, es numérico, inapropiado para las personas. A partir del nivel 4, son simbólicos, es decir, contienen nombres significativos para la gente.
- A este nivel se le denomina **Lenguaje Ensamblador** y es una forma más grata de escribir programas.

9 Firmware y Máquinas Multinivel

Nivel 5

- Este es el nivel de **Lenguajes Orientados a Problemas**, también llamados Lenguajes de Alto Nivel, los cuales son los habitualmente usados por los programadores de aplicaciones.

10 Middleware

- Middleware es software para sistemas distribuidos.
- Permite la interacción entre múltiples procesos que se ejecutan en uno o más computadores en una red.
- Facilita sistemas distribuidos heterogéneos.
- Simplifica la programación de applicaciones.
- Ejemplo, Open DataBase Connectivity (ODBC), que permite a las aplicaciones accesar las bases de datos a través de un middleware llamado driver de ODBC.

GENERACIONES DE MÁQUINA

- **0** (hasta 1945): Mecánicas.
- **1** (1945 – 1955): Tubos de Vacío.
- **2** (1955 – 1965): Transistores.
- **3** (1965 – 1980): Circuitos Integrados.
- **4** (1980 – hoy): VLSI y PC.