



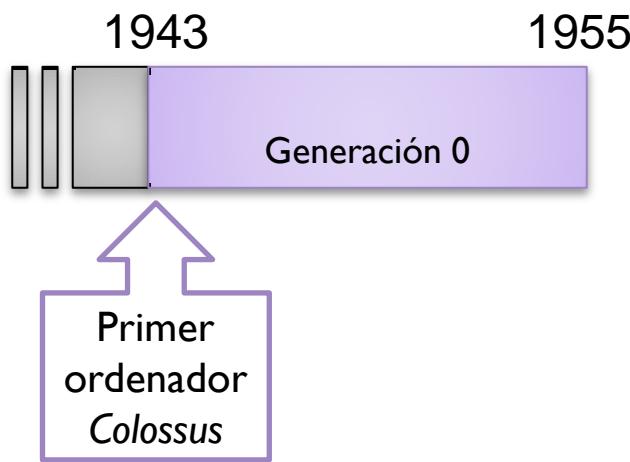
Tema 2

La arquitectura del sistema operativo

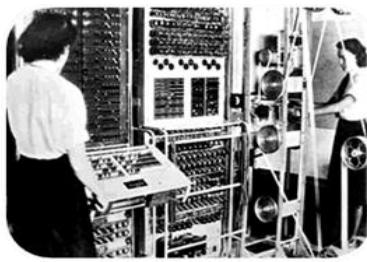
2.1 ¿Qué es un sistema operativo?

- Un sistema operativo (SO o OS) o software base es el conjunto de programas que efectúan la gestión de los recursos de un sistema informático, y permite su normal funcionamiento.
- El sistema operativo es fundamental para un sistema informático. Sin él, el hardware sólo sería una entidad física sin utilidad, ya que no podríamos asignarle ningún tipo de tarea.
- Su funciones principales son:
 - Proporcionar una interfaz entre los usuarios y el hardware.
 - Gestionar los recursos del sistema informático para la óptima ejecución de las tareas:
 - CPU → Gestión de los procesos
 - Memoria principal → Gestión de la memoria
 - Memoria secundaria → Gestión de archivos
 - Periféricos → Gestión de los dispositivos de E/S
 - Red → Gestión de los recursos de red
 - Implementar sistemas de protección.
- Otras funciones del sistema operativo:
 - Proporcionar conectividad a otros sistemas.
 - Tratamiento de errores y seguridad.

2.2 Evolución de los sistemas operativos



El automatismo es muy reducido. El SO es el ser humano



La interacción con el sistema se realiza mediante una consola



Los programadores introducen los programas mediante cableado

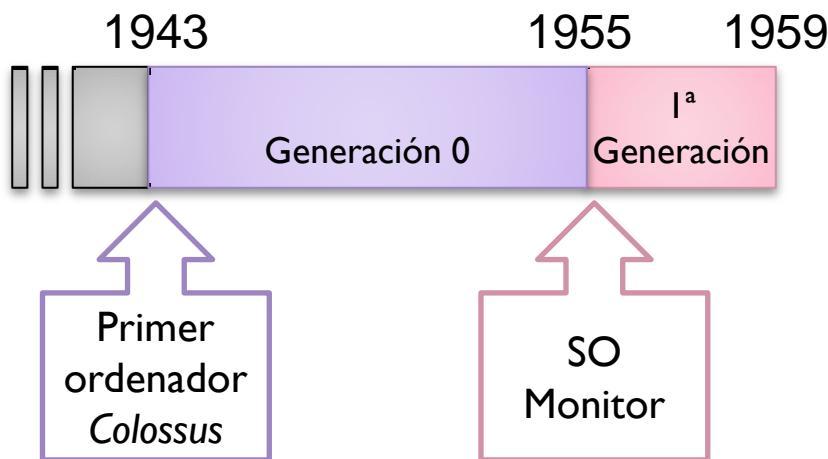


El único periférico de salida es la impresora que muestra los resultados

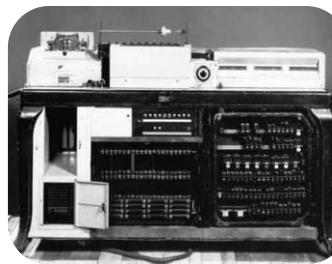
2.2 Evolución de los sistemas operativos

- **Generación cero (1944-1955)**
 - Este modo de funcionamiento es lo que se llama **monoproceso**:
 - Los programas había que cargarlos de uno en uno y manualmente.
 - La ejecución también se efectúa de uno en un proceso (**monoprogramado**).
 - La perdida de eficacia era muy grande debido a que no existía un sistema operativo que planificase ni automatizase la ejecución de los programas.

2.2 Evolución de los sistemas operativos



Tienen un sistema operativo muy básico llamado *Monitor*.



Los programas se introducían con una unidad de carga.



Los programadores introducen mediante tarjetas perforadas



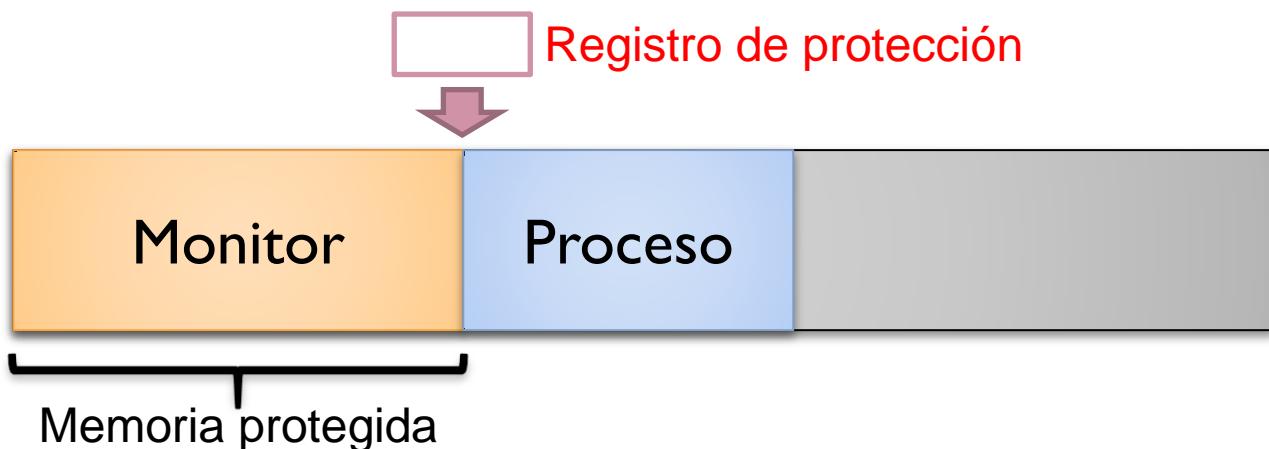
El único periférico de salida sigue siendo la impresora

2.2 Evolución de los sistemas operativos

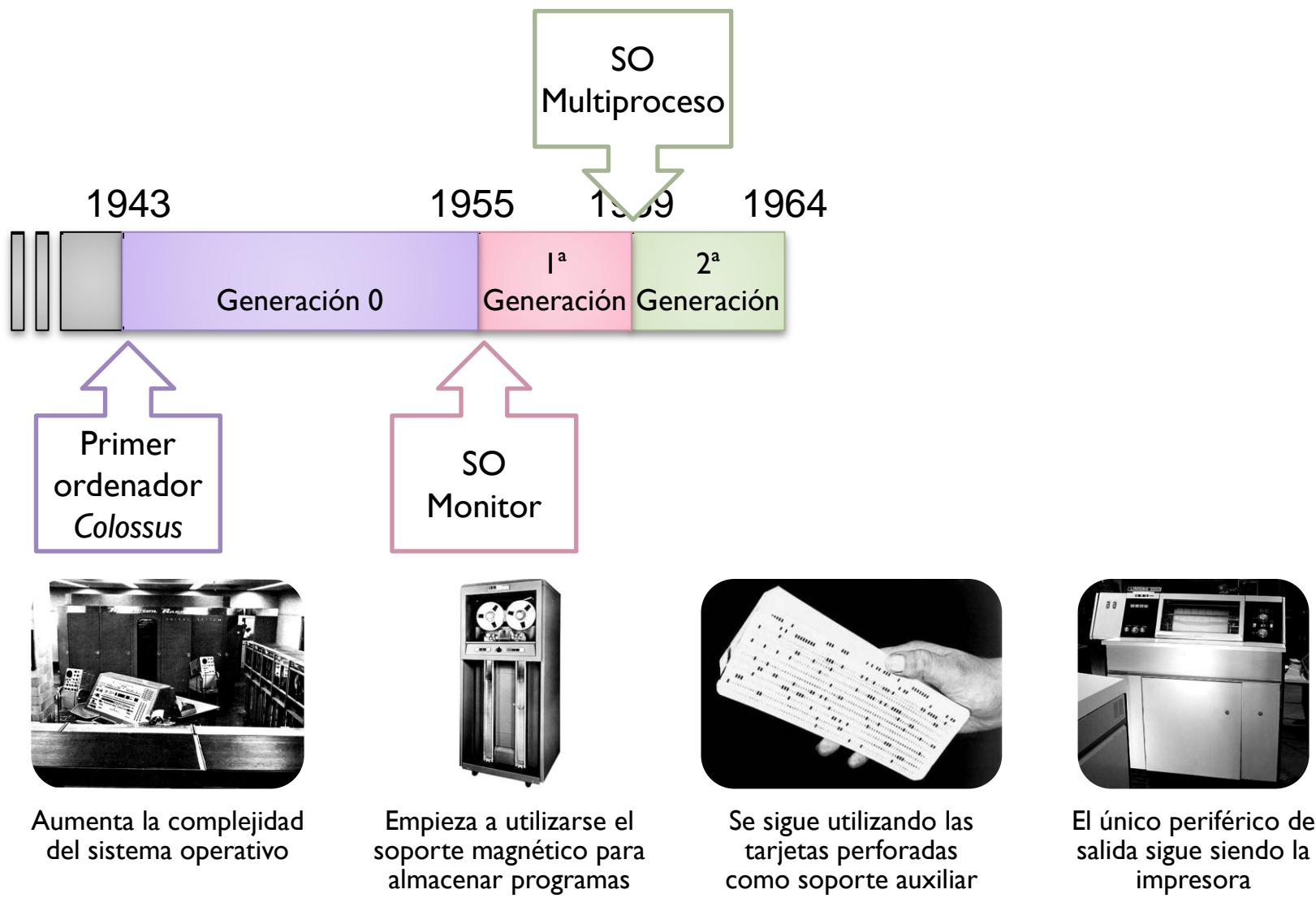
- **Primera generación (1955-1959)**
 - El modo de funcionamiento de estos sistemas era de **procesamiento por lotes**:
 - Los programas están estructurados como secuencias lineales de tareas u ordenes.
 - El *Monitor* se limita a cargar los programas en memoria para ser ejecutados de principio a fin (si no ocurre ningún error o se agota un tiempo de espera).
 - El control del hardware se alterna entre el programa en uso y el *Monitor* (no hay interrupciones).
 - El sistema *Monitor* es lo que llamamos un sistema operativo de **núcleo monolítico**. Es decir un conjunto de rutinas enlazadas entre sí formando un único cuerpo.

2.2 Evolución de los sistemas operativos

- **Primera generación (1955-1959)**
 - El *Monitor* es un sistema operativo que trabaja en modo **Monoprogramado**.
 - Se implementa un primer sistema de **protección de la memoria**:
 - Las direcciones de memoria donde reside el Monitor no se pueden modificar por una aplicación.
 - Si ocurriese el procesador debería detectarlo y devolver el control al *Monitor*.



2.2 Evolución de los sistemas operativos



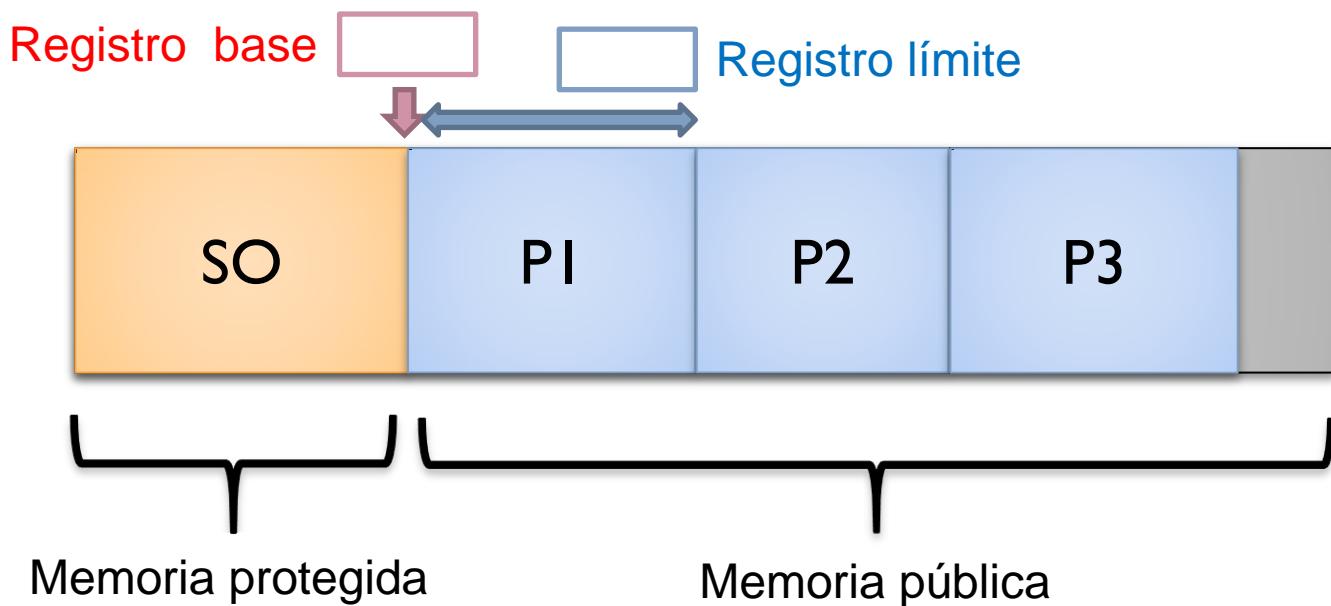
2.2 Evolución de los sistemas operativos

- **Segunda generación (1959-1964)**
 - Aparece el concepto de **multiprogramación** o multitarea.
 - En un SO Multiprogramado *aparentemente* se pueden realizar varias tareas simultáneamente.
 - Pero en realidad se asigna un tiempo finito y muy corto (*quantum*) a cada tarea, alternándose rápidamente y creando la ilusión de simultaneidad.
 - Para implementar la **multiprogramación** en un sistema operativo se necesita necesariamente cargar todos los procesos que se están ejecutando al mismo tiempo en memoria principal.
 - No se pueden ejecutar más procesos de los que permite esta memoria.
 - Aparece el concepto de *Interrupción*.

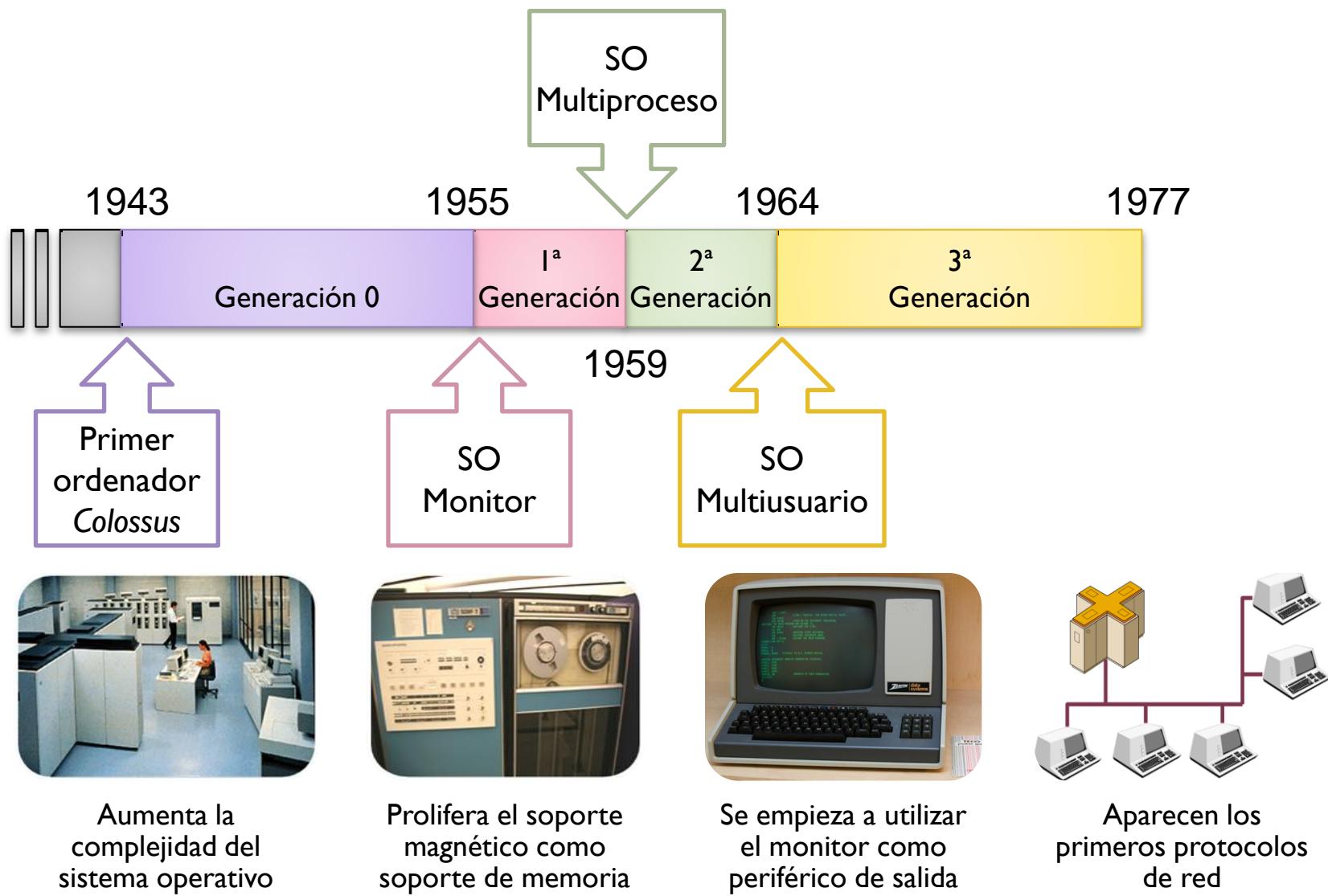


2.2 Evolución de los sistemas operativos

- Segunda generación (1959-1964)



2.2 Evolución de los sistemas operativos



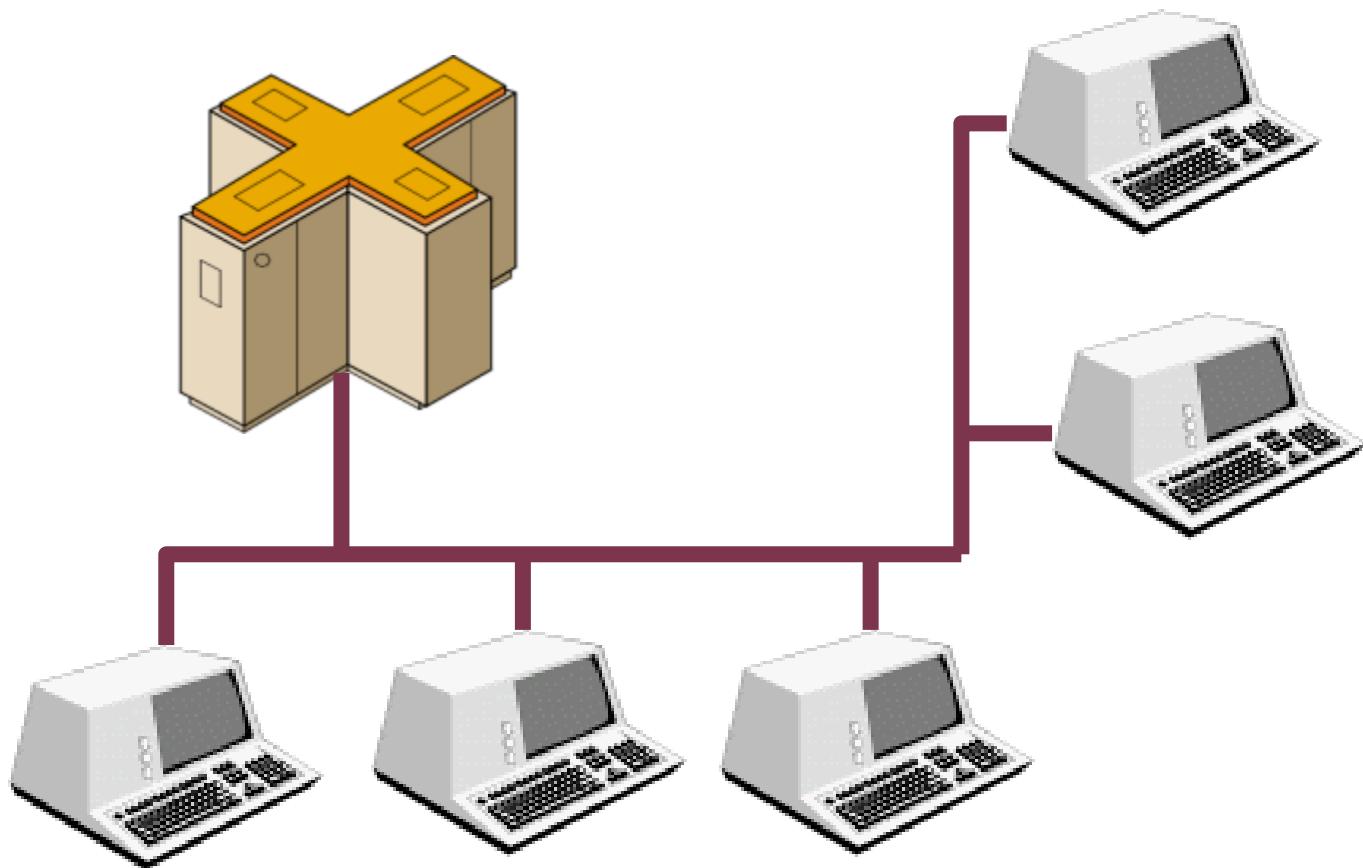
2.2 Evolución de los sistemas operativos

- **Tercera generación (1964-1977)**
 - Es la etapa de los *Mainframes*, ordenadores centrales que controlan un número variable de terminales “tontos”.
 - Aparece el concepto de modo **multiusuario**: Los usuarios tienen la ilusión de que utilizan los recursos simultáneamente.
 - Aparece el concepto de modo **tiempo compartido**: Los usuarios tienen la ilusión de que utilizan los mismos recursos a la vez.
 - La automatización informática llega a todos los estratos de la vida cotidiana (tráfico, aeropuertos, fábricas...)
 - Aparece el concepto de modo **tiempo real**: Se procura que los tiempos de respuesta sean mínimos. Este modo es incompatible con el modo **tiempo compartido**.
 - Los ordenadores de esta etapa son de **propósito general** (multimodo), capaces de trabajar en lotes, en modo multiprogramado, en tiempo real o en tiempo compartido.

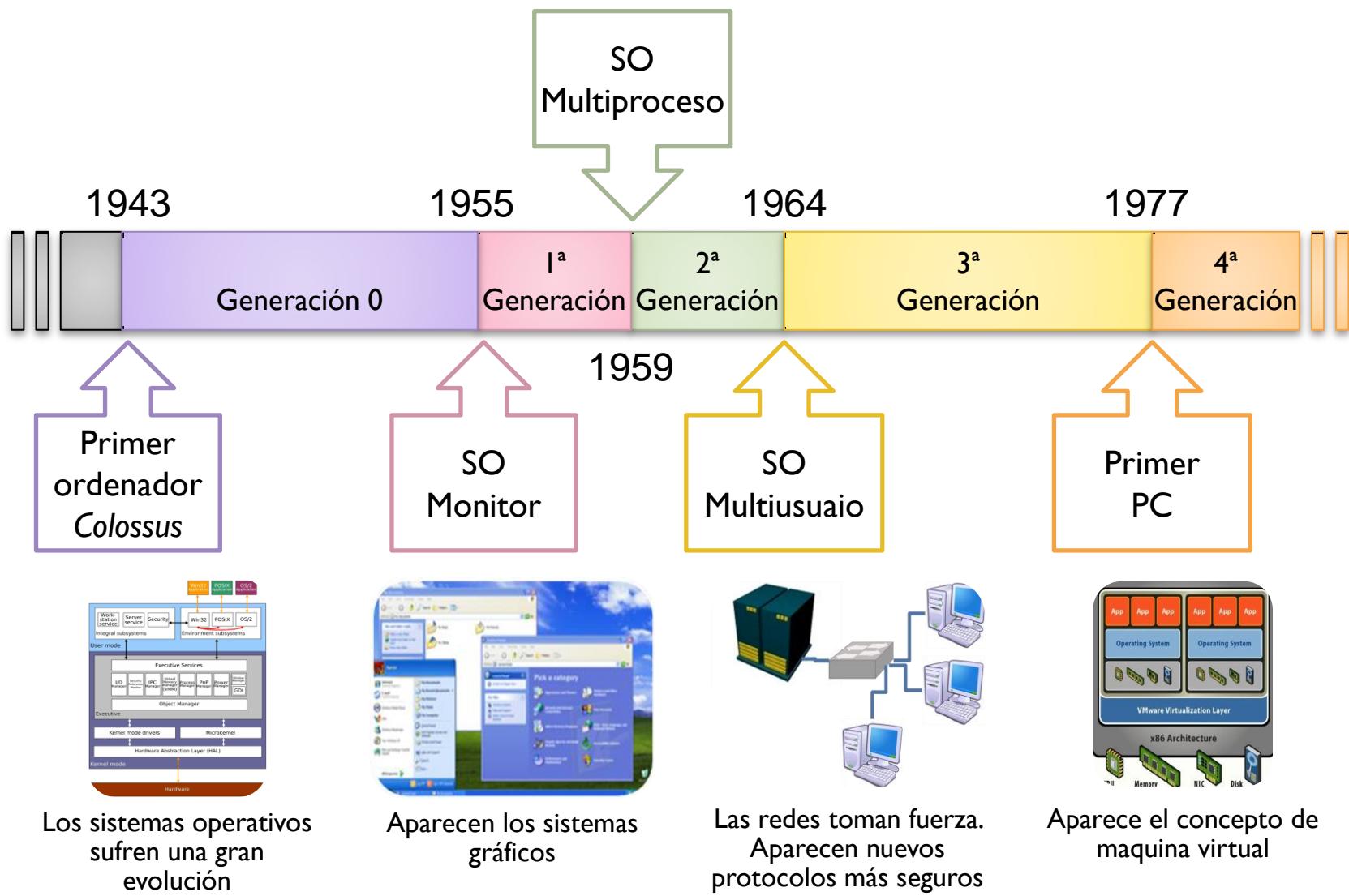


2.2 Evolución de los sistemas operativos

- **Tercera generación (1964-1979)**



2.2 Evolución de los sistemas operativos





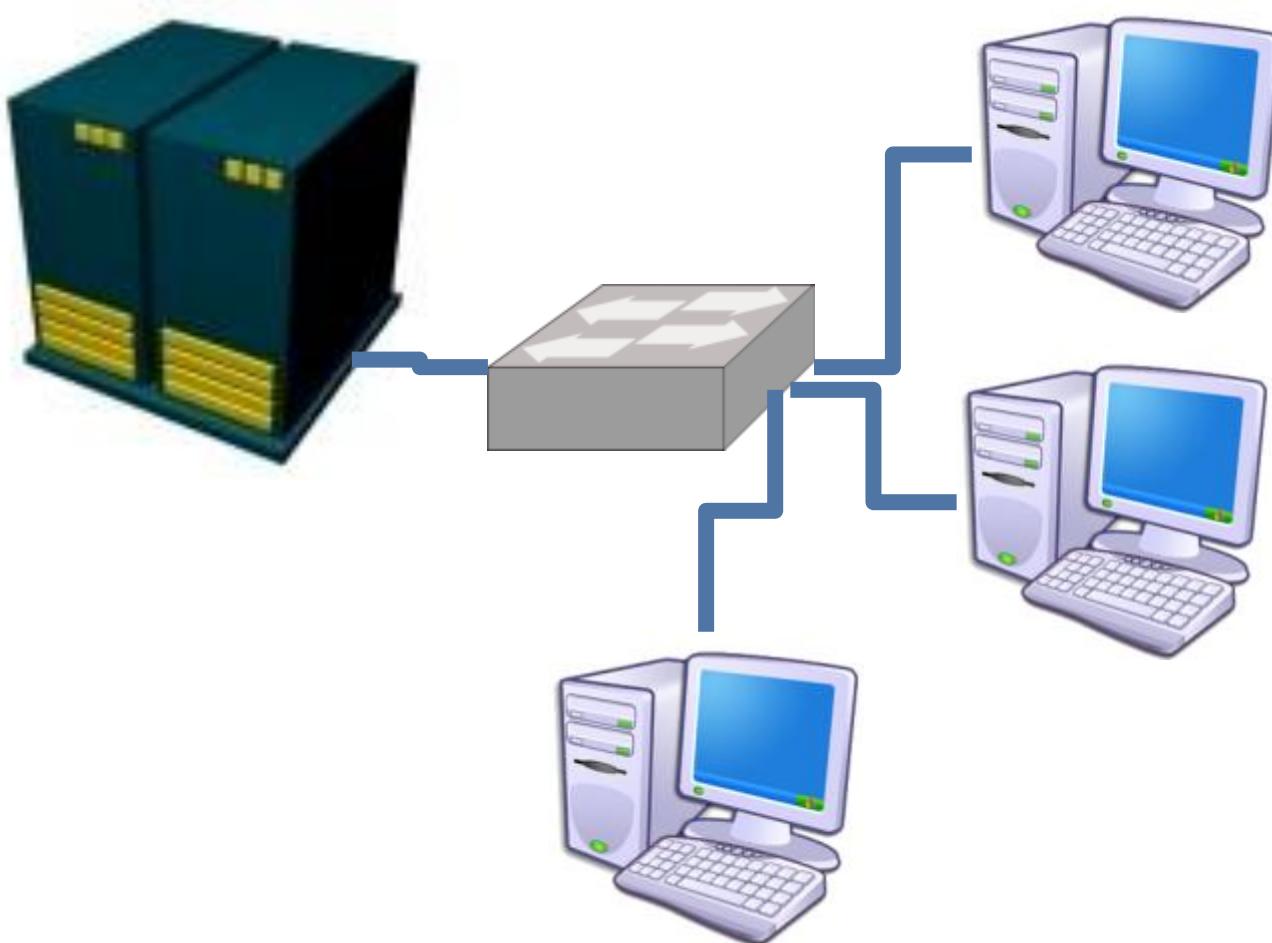
2.2 Evolución de los sistemas operativos

- **Cuarta generación (1977 en adelante)**
 - Aparece el ordenador personal:
 - Los terminales son ahora independientes y computadoras en sí mismas.
 - Tienen menos potencia que los mainframes.
 - Son más accesibles al usuario.
 - Su interfaz es más cómoda y amena → Entorno gráfico.
 - Se idean sistemas operativos capaces de trabajar en una organización en red cliente-servidor.
 - La multiprogramación se hace esencial en los sistemas.
 - Aparecen redes distribuidas.
 - Se fija como objetivo introducir un ordenador en cada casa y conectarlo a la red mundial (Internet).
 - Se multiplica la funcionalidad y versatilidad. Aparece el concepto de multimedia.



2.2 Evolución de los sistemas operativos

- **Cuarta generación (1977 en adelante)**



Sistemas operativos de 4^a generación de Microsoft



Año	Núcleo MS-DOS	Núcleo Windows NT	Núcleo Windows CE
1985	Windows 1.x (16 bits)		
1987	Windows 2.x (16 bits)		
1990	Windows 3.x (16 bits)		
1993		Windows NT (32 bits)	
1995	Windows 95 (16-32 bits)		
1996			Windows CE 1.x (32 bits/ARM)
1997			Windows CE 2.x (32 bits/ARM)
1998	Windows 98 (16-32 bits)		
2000	Windows Me (16-32 bits)	Windows 2000 (32 bits)	Windows CE 3.x (32 bits/ARM)
2001		Windows XP (32/64 bits)	
2002			Windows CE 4.x (32 bits/ARM)
2003		Windows 2003 Server	
2004			Windows CE 5.x (32 bits/ARM)
2006			Windows CE 6.x (32 bits/ARM)
2007		Windows Vista (32/64 bits)	
2008		Windows 2008 Server	
2009		Windows 7 (32/64 bits)	
2010			
2011			Windows CE 7.x (32 bits/ARM)
2012		Windows 8 (32/64 bits/ARM)	



1965 UNIX

1983 HP-UX

1985 SunOS/Solaris/OpenSolaris

1986 AIX (IBM)

1978 BSD

1993 FreeBSD

1993 NetBSD

1995 OpenBSD

2000 Mac OS X

1987 MINIX

1991 Linux

1993 Debian

1993 Red Hat/Fedora

1993 SlackWare/SuSE/OpenSuSE

2004 Ubuntu



Sistemas operativos de 4^a generación de Apple

Año	Familia Lisa	Familia Macintosh	Familia Macintosh basado en BSD
1982	Lisa OS		
1984	Lisa OS II	Mac System 1.x	
1985		Mac System 2.x	
1986		Mac System 3.x	
1987		Mac System 4.x Mac System 5.x	
1988		Mac System 6.x	
1991		Mac System 7.x	
1997		Mac OS 8	
1999		Mac OS 9	Mac OS X Server 1.0 Rhapsody
2000			Mac OS X Beta Kodiak /Server
2001			Mac OS X 10.0 Cheetah /Server Mac OS X 10.1 Puma /Server
2002			Mac OS X 10.2 Jaguar /Server
2003			Mac OS X 10.3 Panther /Server
2005			Mac OS X 10.4 Tiger /Server
2007			Mac OS X 10.5 Leopard /Server
2009			Mac OS X 10.6 Snow Leopard /Server
2011			Mac OS X 10.7 Lion /Server
2012			Mac OS X 10.8 Mountain Lion /Server
2013			Mac OS X 10.9 Mavericks /Server

Sistemas operativos de 4^a generación en dispositivos móviles

Año	Windows CE	Windows NT	Apple iOS	Nokia/Accenture
1989				EPOC16
1997				EPOC32 1.x
1998				EPOC32 4.x
1999				EPOC32 5.x
2000	PocketPC 2000			
2001				Symbian OS 6.x
2002	Pocket PC 2002			
2003	Windows Mobile 2003			Symbian OS 7.x
2004				Symbian OS 8.x
2005	Windows Mobile 5.x			
2006				Symbian OS 9.x
2007	Windows Mobile 6.x		iOS 1.x	
2008			iOS 2.x	
2009			iOS 3.x	
2010	Windows Phone 7.x		iOS 4.x	
2011			iOS 5.x	Symbian OS 10.x
2012		Windows Phone 8.x Windows RT	iOS 6.x	
2013			iOS 7.x	

Sistemas operativos de 4^a generación en dispositivos móviles

Año	BlackBerry (Java)	BlackBerry (QNX)	Samsung (QNX)	Palm/HP/LG (Linux)
1999	BlackBerry OS 1.x			
2001				
2002	BlackBerry OS 3.x			
2003				
2004				
2005	BlackBerry OS 4.x			
2006				
2007				
2008	BlackBerry OS 5.x			
2009				WebOS 1.x
2010	BlackBerry OS 6.x	BlackBerry Tablet OS 1.x	Bada 1.x	WebOS 2.x
2011	BlackBerry OS 7.x		Bada 2.x	WebOS 3.x
2012		BlackBerry Tablet OS 2.x		
2013		BlackBerry 10.x		

Sistemas operativos de 4^a generación en dispositivos móviles

Año	Google (Linux)	Nokia (Linux)	Intel (Linux)	Samsung (Linux)
2005		Maemo 1.x		
2006		Maemo 2.x		
2007		Maemo 3.x	Moblin 1.x	LiMo 1.x
2008	Android 1.0 Apple Pie	Maemo 4.x		
2009	Android 1.1 Banana Bread Android 1.5 Cupcake Android 1.6 Donut Android 2.0 Eclair	Maemo 5.x	Moblin 2.x	LiMo 2.x
2010	Android 2.2 Froyo Android 2.3 Gingerbread	MeeGo 1.x		
2011	Android 3.x Honeycomb Android 4.0 Ice Cream Sandwich			LiMo 4.x
2012	Android 4.1 Jelly Bean Android 4.2 Gummy Bear Android 4.3 Gominola		Tizen 1.x	
2013	Android 4.4 KitKat Android 5.0 Key Lime Pie		Tizen 2.x Tizen 3.x	

2.2 Evolución de los sistemas operativos

- **¿Quinta generación??**
 - Japón anunció en 1979 la intención de desarrollar sistemas operativos de quinta generación.
 - A este esfuerzo se le sumaron EEUU y otros países europeos.
 - La idea era crear *inteligencia artificial* (AI) que emulase el pensamiento humano.
 - El proyecto fue un rotundo fracaso y después de muchas decepciones, se abandonó en 1995.
 - Algunos de sus logros están hoy en día en los ordenadores de cuarta generación.
 - Integración de las redes inalámbricas.
 - Reconocimiento de voz.
 - Reconocimiento de imágenes.

2.2 Evolución de los sistemas operativos

- **Dispositivos móviles y la 3^a generación**
 - Hoy, poco a poco, principalmente en dispositivos móviles, vuelven a las filosofías adoptadas en la tercera generación.
 - Los nuevos sistemas operativos tienen a ser más ligeros y eficientes, optimizados para un hardware de baja prestaciones.
 - Los sistemas cada vez más delegan sus recursos de computo o de almacenaje a servidores externos llamadas *nubes de computación*.
 - Esto produce un abaratamiento de los sistemas informáticos hasta tal punto de volver a ser meros terminales.
 - Cobra muchísima importancia la estabilidad y el ancho de banda de las conexión a Internet.

2.2 Evolución de los sistemas operativos

- **Modos de explotación actuales**
 - **Modos según el número de procesos simultáneos:**
 - **Modo Monoprogramado:** El sistema ejecuta los procesos de uno en uno.
 - **Modo Multiprogramado:** El sistema crea la ilusión de que varios procesos se ejecutan simultáneamente.

SO Monoprogramados	SO Multiprogramados
Monitor Ms-DOS	Sistemas Windows Sistemas UNIX Distribuciones Linux Sistema Lisa Sistemas Mac X/iOS Sistemas Symbian Sistemas Blackberry Sistemas Middleware QNX Sistemas Middleware Linux BeOS

2.2 Evolución de los sistemas operativos

- **Modos de explotación actuales**
 - **Modos según el número de usuarios:**
 - **Modo Monousuario:** Un único usuario utiliza los recursos del sistema en un mismo instante.
 - **Modo Multiusuario:** Varios usuarios pueden utilizar los recursos del sistema simultáneamente.

SO Monousuarios	SO Multiusuarios
Windows XP Windows 7 Windows 8 Familia Windows CE Sistemas iOS Sistemas Symbian Sistemas Blackberry Sistemas Middleware QNX Sistemas Middleware Linux BeOS	Windows 2003 Server Windows 2008 Server Sistema UNIX Distribuciones Linux Sistemas Mac X

2.2 Evolución de los sistemas operativos

- **Modos de explotación actuales**
 - **Modos según el tiempo de respuesta:**
 - **Tiempo compartido:** Cuando lo prioritario es que cada usuario disponga de los recursos de forma equitativa. Propio de los ordenadores centrales (Mainframes).
 - **Tiempo real:** Cuando lo prioritario es que la respuesta sea lo más rápida posible, antes de sobrepasar un umbral de tiempo.

SO de Tiempo Real	SO de Tiempo Compartido
Familia Windows CE Sistemas iOS Sistemas Symbian Sistemas Blackberry Sistemas Middleware QNX Sistemas Middleware Linux BeOS	Windows 200x Server Sistema UNIX Distribuciones Linux Sistemas Mac X

2.2 Evolución de los sistemas operativos

- **Modos de explotación actuales**
 - **Modos según el número de procesadores:**
 - **SO Monoprocesador:** Sólo hay un procesador. Y por tanto sólo se puede ejecutar un proceso al mismo tiempo.
 - **SO Multiprocesador:** Varios procesadores ejecutan simultáneamente cauces de proceso diferentes.
 - **Arquitectura NUMA (Multiprocesamiento Asimétrico):** Los procesadores disponen de regiones de memoria aisladas con buses de datos diferentes. Normalmente hay un procesador maestro (*master*) que delega funciones y asigna recursos a los demás esclavos (*slaves*).
 - **Arquitectura UMA (Multiprocesamiento Simétrico):** Los procesadores comparten un único espacio de memoria y un bus de datos único.
 - Multiprocesamiento en chips diferentes (SMP): Los procesadores sólo comparten la memoria y el bus de datos.
 - Multiprocesamiento a nivel de chip (CMP): Los procesadores comparten memoria, buses, cache y otros recursos.

2.2 Evolución de los sistemas operativos

- **Propósito de los sistemas operativos**
 - **SO para procesamiento por lotes**: Diseñados para el procesamiento continuo de tareas.
 - **SO para servidores**: diseñados para dar servicios a otros ordenadores. Gestionan un amplio ancho de transmisión.
 - **SO para ordenadores personales**: pensados para usuarios de PC con pocos conocimientos informáticos.
 - **SO para dispositivos móviles**: pensados para tabletas y teléfonos móviles.
 - **SO embebidos**: se encuentran en el firmware de dispositivos electrónicos menores, son muy sencillos y no permiten la reconfiguración.
 - **SO para tarjetas inteligentes**: se encuentran embebidos en chips de tarjetas TCI.

2.2 Evolución de los sistemas operativos

- Propósito de los sistemas operativos

Servidores	Clients	Móviles	Embebidos	TCI
Windows 2003 Server	Windows XP	Windows Mobile	Windows CE	Java Card
Windows 2008 Server	Windows 7	Windows Phone	QNX	
Windows 2012 Server	Windows 8	Sistemas iOS		
Sistema UNIX	Distribuciones Linux	Sistemas Symbian		
Sistemas Mac X	Sistemas Mac X	Sistemas Blackberry		
Distribuciones Linux	BeOS	Middleware QNX Middleware Linux		

2.2 Evolución de los sistemas operativos

- **Modos de explotación actuales**

Monoprogramación	Procesamiento por lotes	Monousuario	Tiempo Real
Multiprogramación	Tarjetas TCI		Multimodo
	Embebidos		
	Móviles		
	Clientes		Tiempo Compartido
	Servidores	Multiusuario	
	Mainframes		

Monoprocesador	
Multiprocesador	Arquitectura Asimétrica (NUMA)
	Arquitectura Simétrica (UMA)
	SMP
	CMP

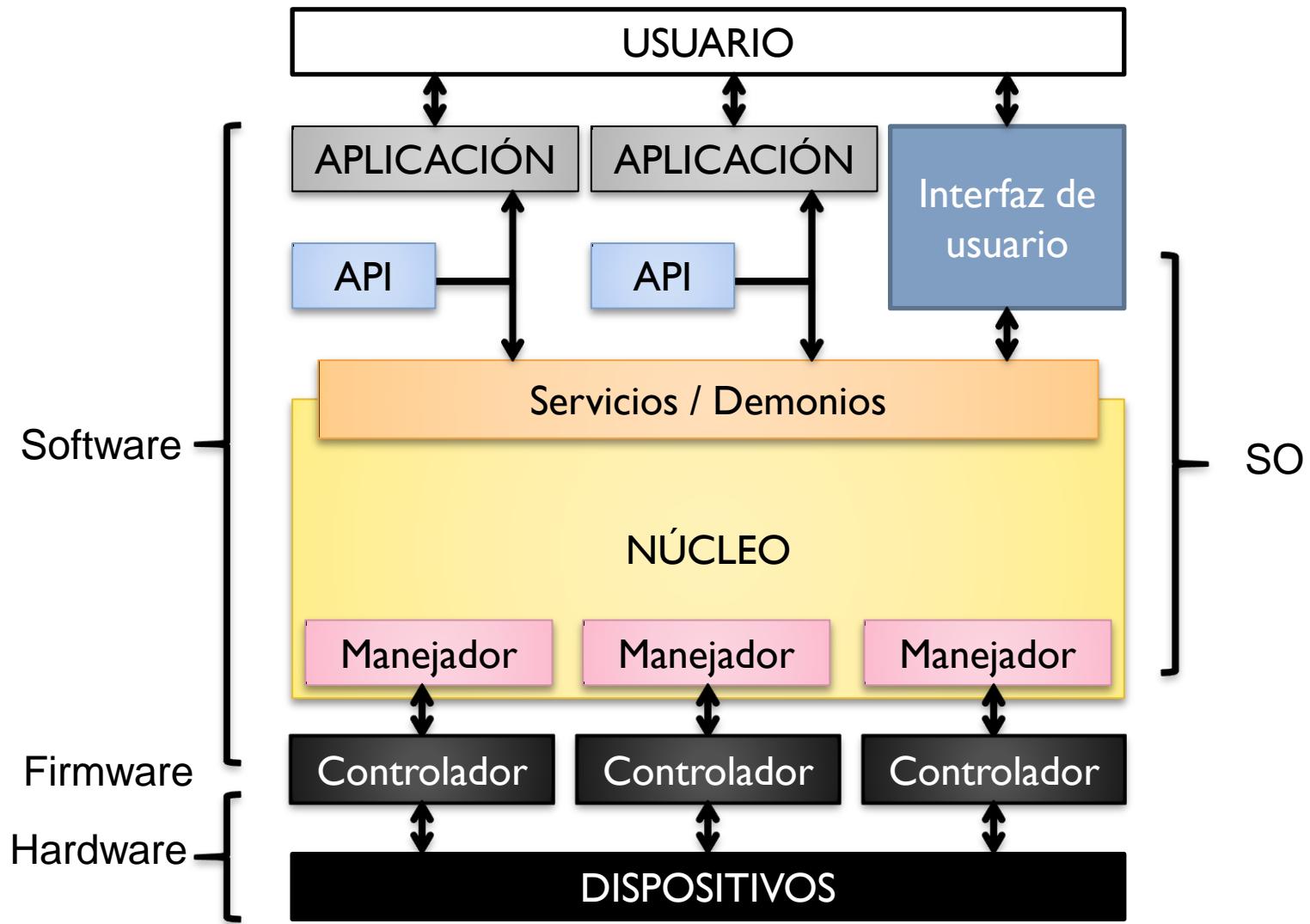


2.3 Estructura de un sistema operativo

- Hacer posible la interacción entre el hardware y el usuario.
 - Se establece un sistema por capas:
 1. **Interfaz de usuario:** Permiten al usuario comunicarse con el sistema operativo.
 - Línea de comandos (CLI) – Modo Texto / Símbolo de Sistema
 - Entorno gráfico (GUI) – Modo Gráfico
 2. **Interfaz de aplicación (API):** Permite a las aplicaciones solicitar recursos al sistema operativo (llamadas al sistema).
 3. **Servicios/Demonios:** Procesos no controlados por el usuario. Gestionan el hardware ejecutándose infinitamente.
 4. **Núcleo:** Controla el funcionamiento de todo el sistema operativo, es quién realiza la gestión de los recursos.
 5. **Manejadores (Drivers):** Cada uno permite la comunicación del sistema operativo con un elemento del hardware concreto.



2.3 Estructura de un sistema operativo





2.3 Estructura de un sistema operativo

• Modo protegido

- En un sistema informático es muy importante limitar el acceso al usuario de ciertos recursos al ser estos *muy delicados*.
- En modo normal o **modo usuario**, lo normal es que no se puedan realizar todas las tareas. Este modo es el que utilizan las llamadas a sistema de las aplicaciones y las solicitudes del usuario.
- En modo protegido o **modo kernel**, se tiene prioridad absoluta y se puede realizar cualquier tarea. Sólo el núcleo del sistema operativo puede ejecutar instrucciones en modo kernel. Este modo tiene acceso directo al hardware.



2.3 Estructura de un sistema operativo

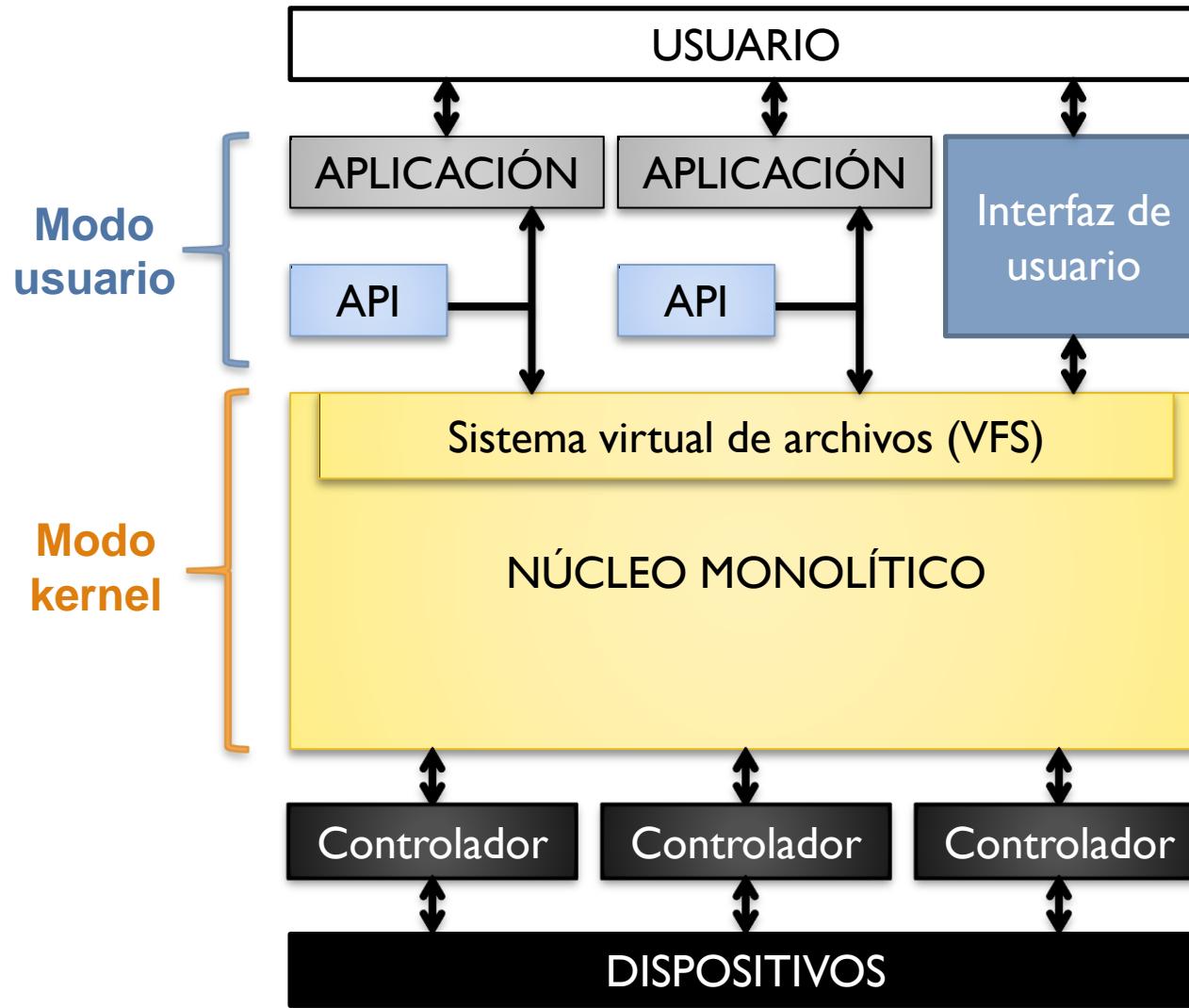
- **Arquitecturas del núcleo**

- **Núcleo monolítico**

- Está compuesto de un único módulo muy complejo que integra todas las funciones del sistema operativo.
 - La comunicación entre sus rutinas internas es total lo que hace más fluido su ejecución.
 - Por otra parte un error puede propagarse a todas las funciones.
 - Estos núcleos ofrecen a las aplicaciones de usuario una interfaz que virtualiza la mayoría de los aspectos (como el de un sistema de archivos virtual), esto otorga al sistema una cierta estabilidad.
 - Cualquier módulo adicional del sistema operativo se ejecuta en el espacio de memoria del núcleo, y por tanto en **modo kernel**.
 - Cada vez que el núcleo se actualice o se modifique es necesario recompilarlo entero. Esto hace muy complicado el diseño e instalación de nuevos manejadores (*drivers*).

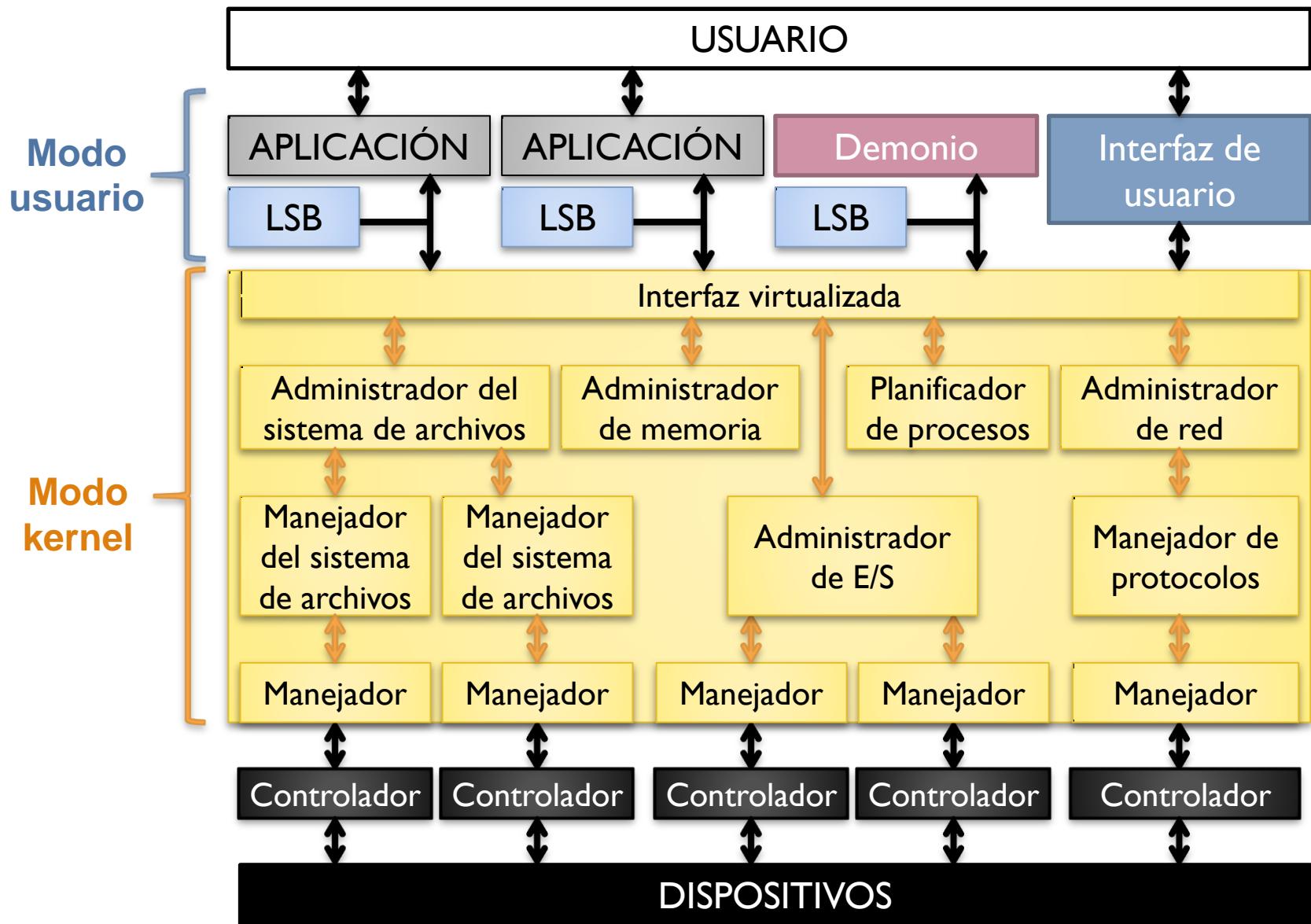


2.3 Estructura de un sistema operativo





Linux





2.3 Estructura de un sistema operativo

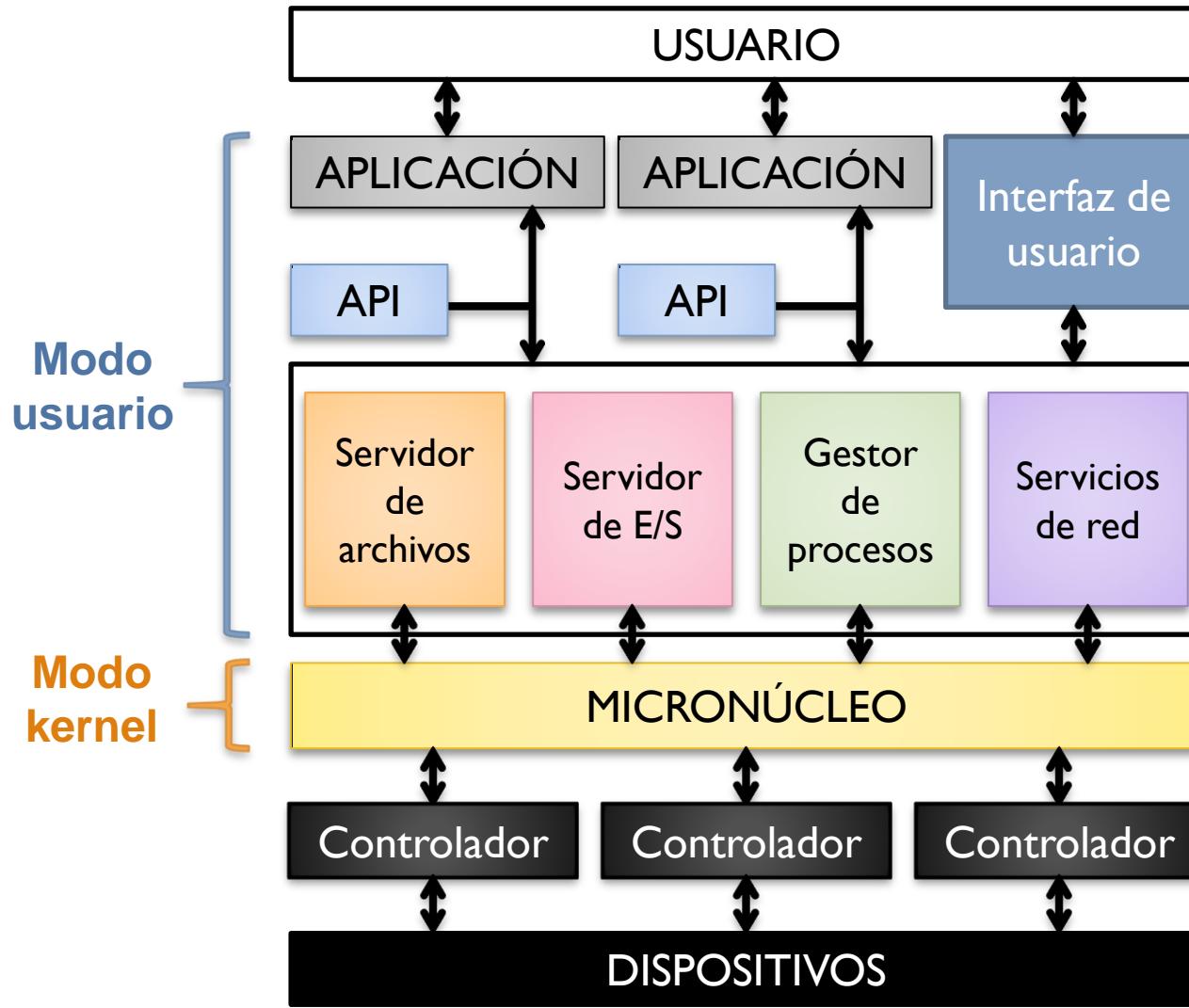
- **Arquitecturas del núcleo**

- **Micronúcleo o microkernel**

- Está compuesto de un conjunto de rutinas muy básicas con las funciones fundamentales del sistema operativo.
 - El resto de funciones se realizan mediante procesos servidores en **modo usuario**.
 - La modularización del sistema operativo lo hace muy resistente a errores al quedarse estos aislados en la parte afectada.
 - La comunicación entre las partes es dificultosa lo que hace de estos sistemas menos eficientes que los monolíticos.
 - La actualización del sistema es sencilla y los nuevos manejadores fáciles de diseñar e instalar.



2.3 Estructura de un sistema operativo





2.3 Estructura de un sistema operativo

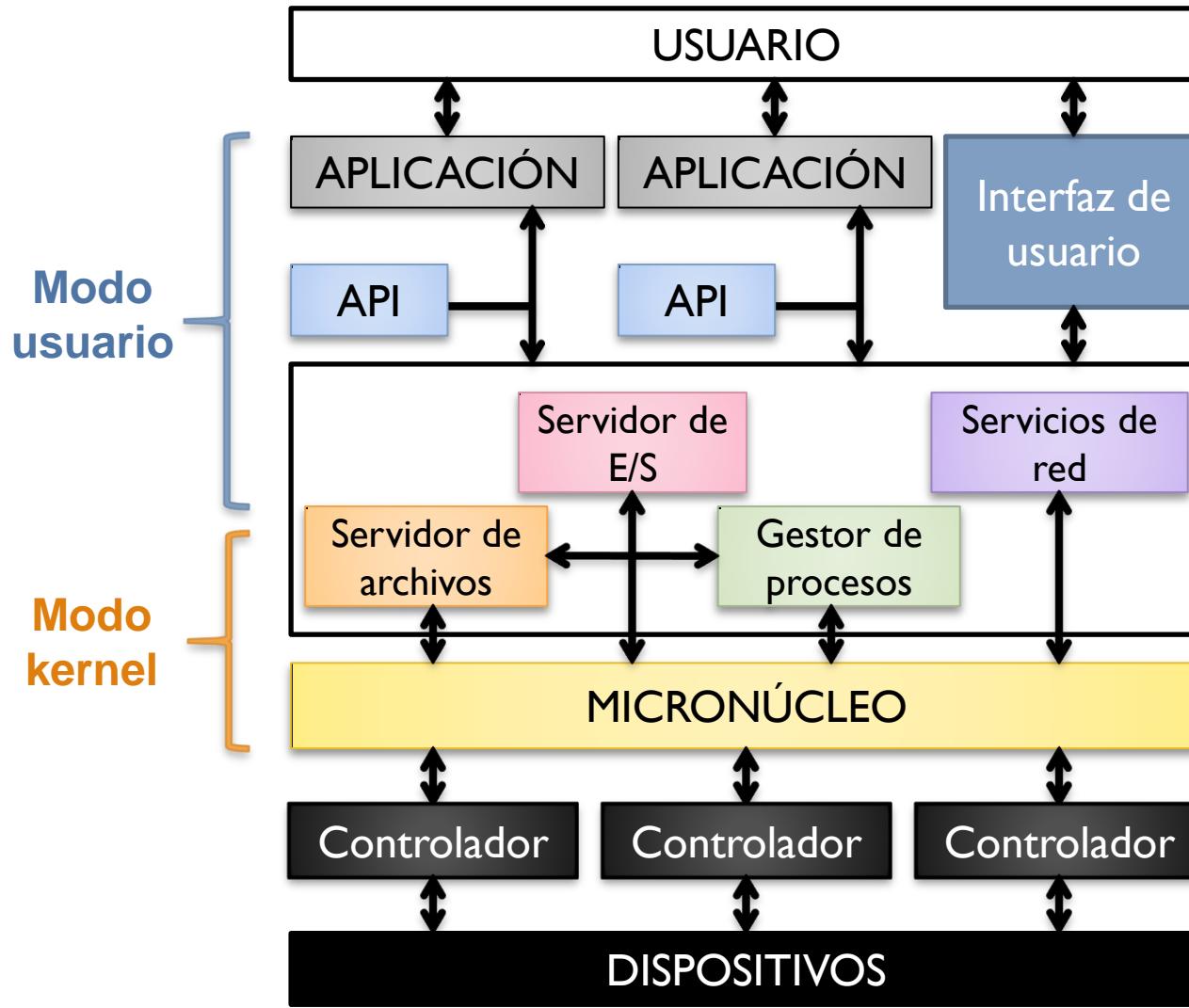
- **Arquitecturas del núcleo**

- **Núcleo híbrido**

- Fundamentalmente es muy parecido a un micronúcleo puro, aunque en este caso integra más funcionalidades que éste.
 - Vamos a tener pues algunos módulos que se van a ejecutar en **modo kernel** y otros en **modo usuario**.
 - Combina los aspectos de los núcleos monolíticos con los de los micronúcleos.
 - La estabilidad del sistema es como la de un micronúcleo, al estar modularizado.
 - La comunicación entre las partes es difícil sólo en **modo usuario**.
 - La actualización del sistema es sencilla y los nuevos manejadores fáciles de diseñar e instalar.

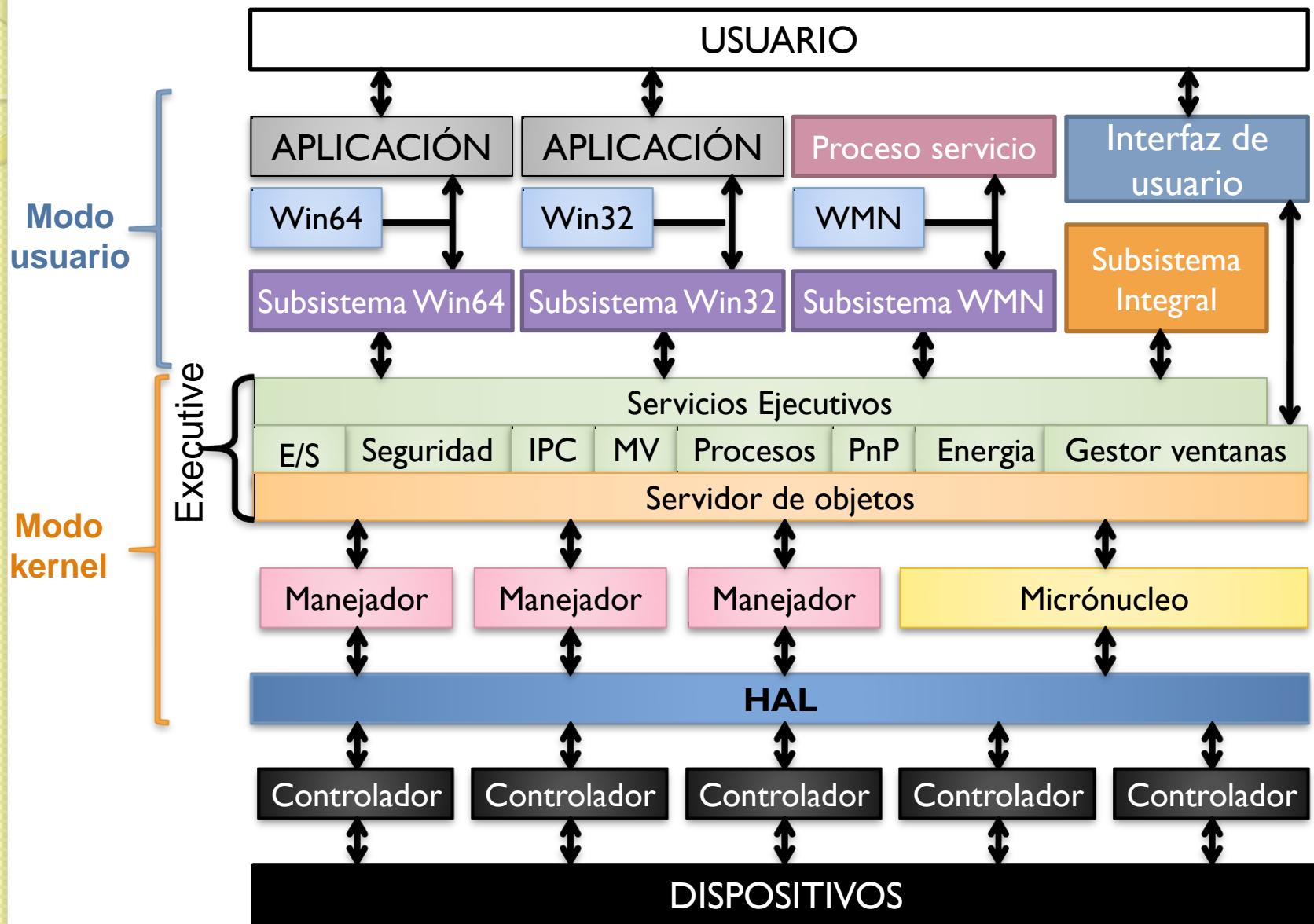


2.3 Estructura de un sistema operativo





Windows NT





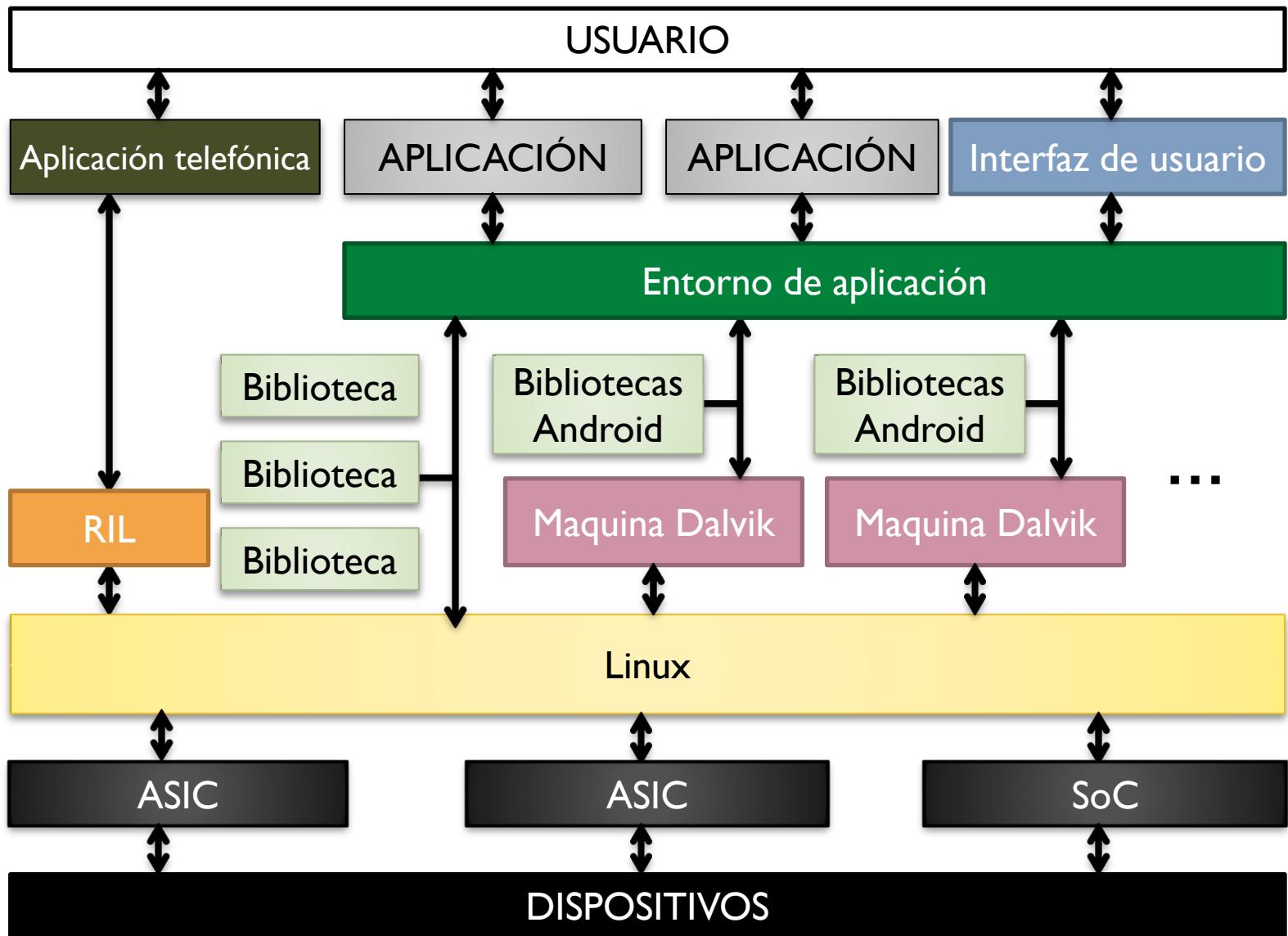
2.3 Estructura de un sistema operativo

- **Middleware**

- Se utiliza una plataforma intermedia entre el núcleo y las aplicaciones, constituyendo una capa de abstracción intermedia.
- El núcleo de estos sistemas suele ser un núcleo monolítico reducido.
- Las aplicaciones se ejecutarán total o en su mayoría mediante un entorno de ejecución (*framework*).
- El nuevo nivel de abstracción proporciona una mayor estabilidad.
- La implementación de aplicaciones se realiza según las especificaciones del entorno de ejecución.
- La actualización de las aplicaciones es sencilla.
- El uso de maquinas virtuales y de librerías nativas posibilita la ejecución multiplataforma.



Android





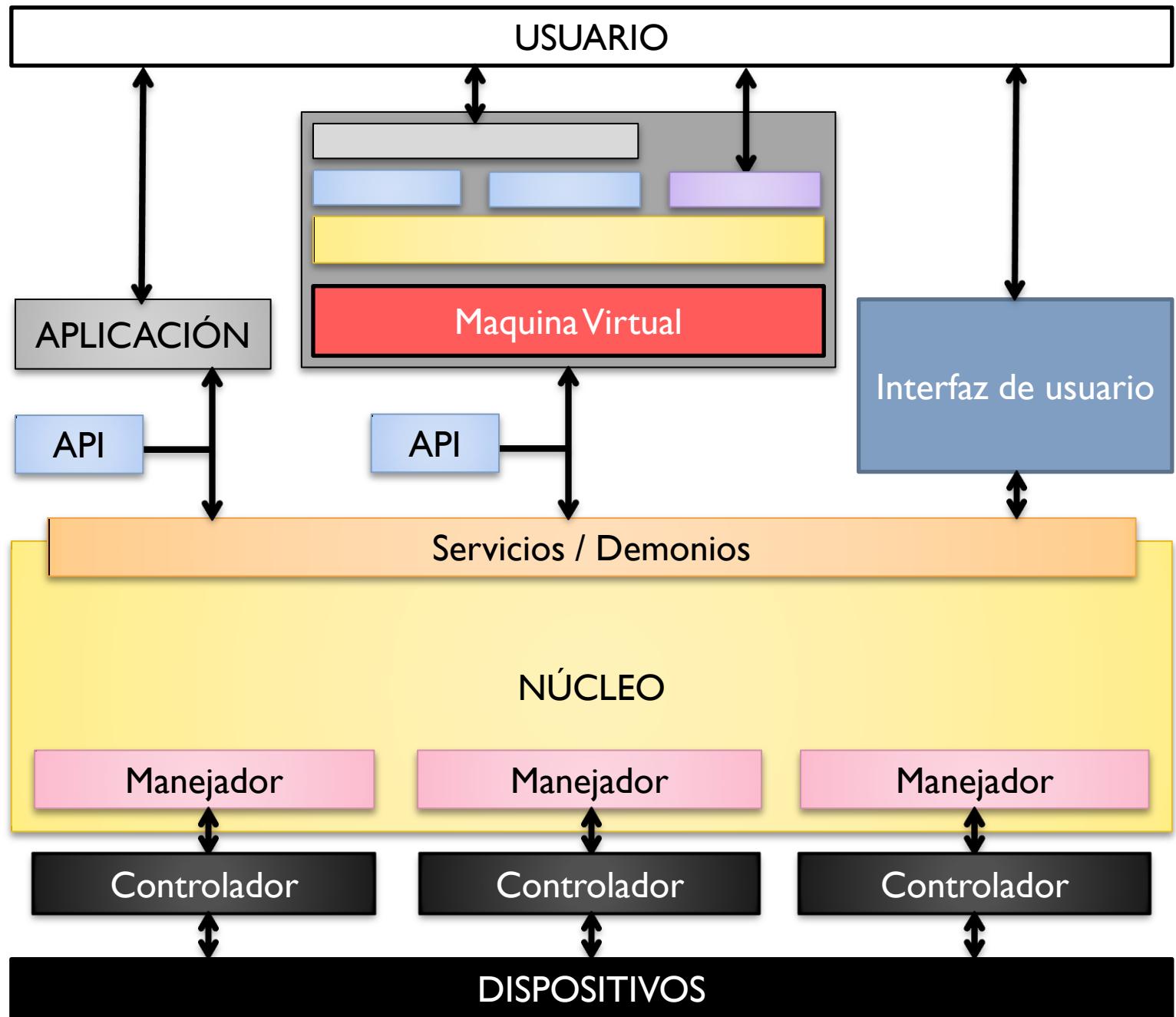
2.3 Estructura de un sistema operativo

Núcleo monolítico	Micronúcleo	Núcleo Híbrido	Middleware
Sistemas UNIX	QNX	Windows XP	Blackberry 10
Distribuciones Linux	BeOS	Windows 7	Samsung Bada
Windows CE	Blackberry OS	Windows 8	LG WebOS
	Symbian	Windows 2003 Server	Android
		Windows 2008 Server	Maemo
		Windows 2012 Server	Moblin
		Windows RT	LiMo
		Windows Mobile	MeeGo
		Windows Phone	Tizen
		Mac OS X	
		iOS	



2.4 Maquinas virtuales

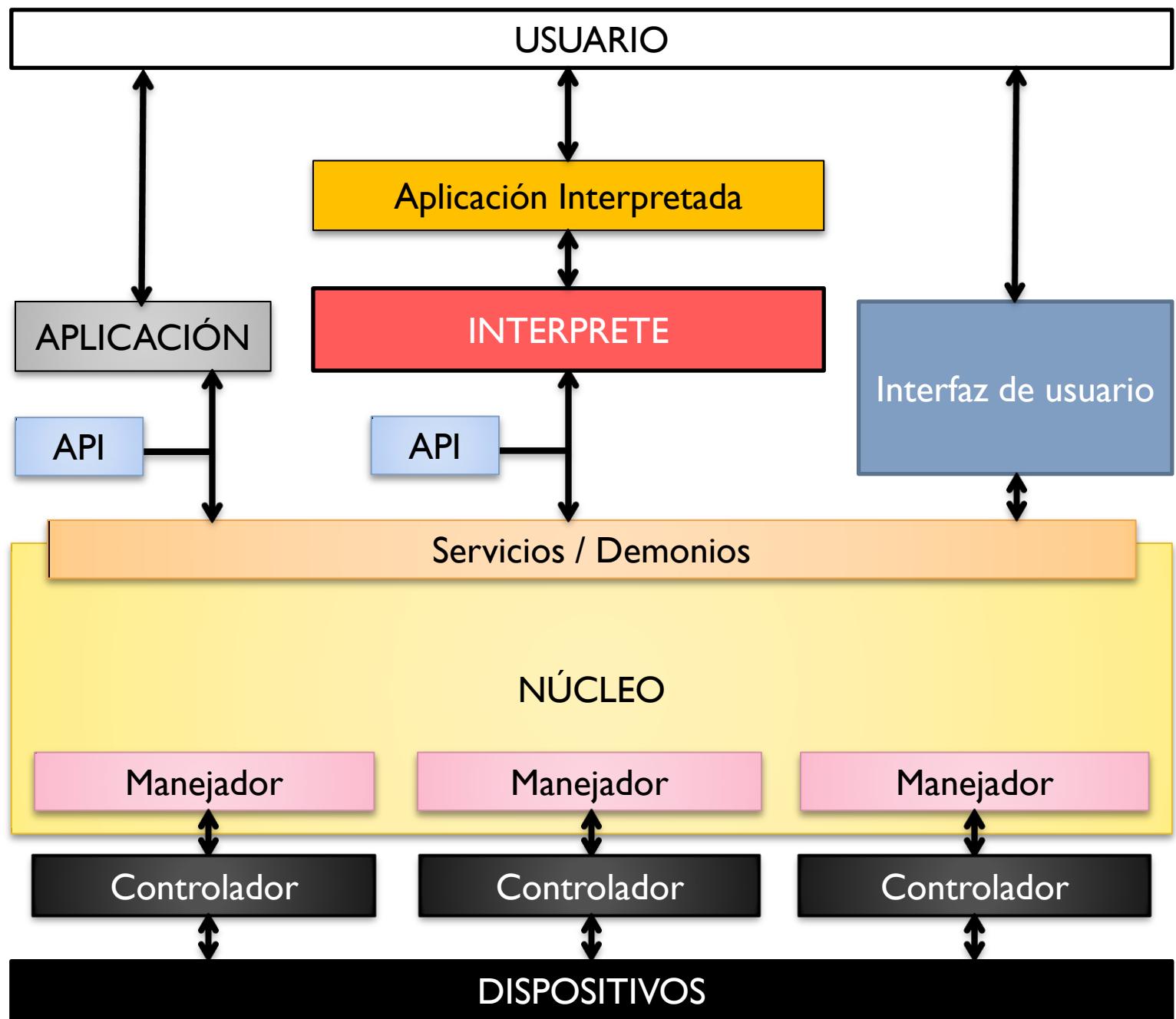
- **Máquinas de ejecución nativa:**
 - Son aplicaciones que emulan el funcionamiento de un ordenador utilizando el hardware y software de un ordenador *anfitrión*.
 - La máquina virtual simula todo el hardware al detalle, aunque éste en realidad no existe.
 - Sobre este hardware simulado se puede instalar cualquier sistema operativo compatible, exactamente de la misma forma que haríamos con una máquina real.
 - El sistema operativo que da soporte a la máquina virtual se llama igualmente **SO Anfitrión**, mientras que el sistema operativo emulado se le llama **SO Invitado**.
 - El Anfitrión provee de todos los recursos que solicita el Invitado mediante **llamadas al sistema** a través de la máquina virtual.
 - La máquina virtual jamás puede tener más potencia que la máquina real.
 - **Ejemplos:**
 - Virtualbox
 - VMware
 - Virtual PC





2.4 Maquinas virtuales

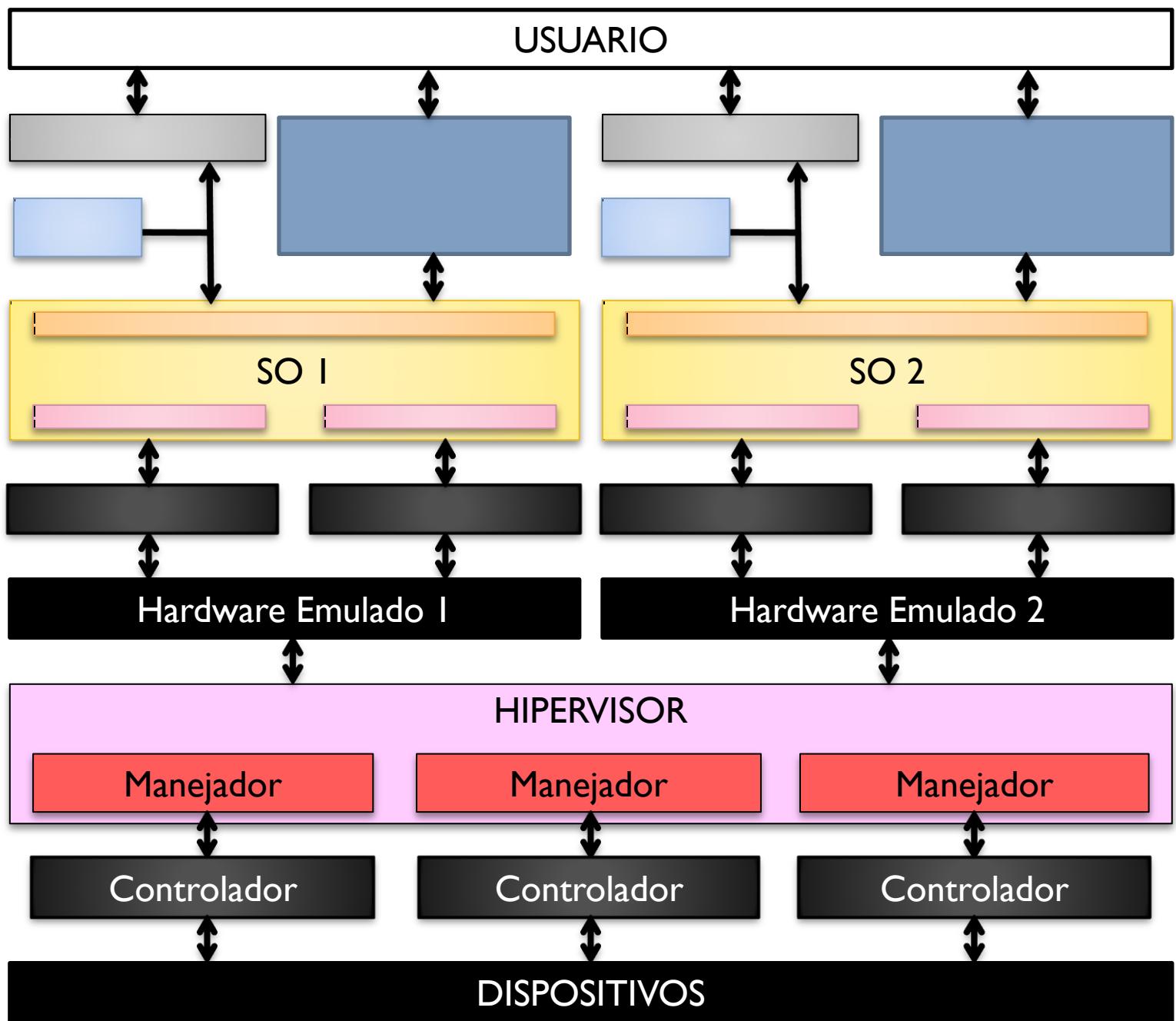
- **Máquinas virtuales de proceso:**
 - Son aplicaciones que crean un entorno específico tanto en hardware como en software para un único proceso.
 - La máquina virtual se ejecuta como una aplicación más.
 - También se emula todas las señales de un ficticio sistema operativo.
 - El hardware emulado es totalmente ficticio.
 - Tipos:
 - **Interpretes** → Trabajar con el código fuente de los programas.
 - Java Virtual Machine (JVM) → Java
 - Common Language Runtime (CLR) → C# ,Visual Basic .NET
 - Macromedia Flash Player → SWF
 - Scumm VM → Juegos de MsDOS
 - **Emuladores** → Trabajan directamente con archivos binarios.
 - DOSBox → Ejecutables de MsDOS
 - Las llamadas **capas de compatibilidad** no son maquinas virtuales sino API no nativas.
 - WINE → Ejecutables de MsDOS, Win16,Win32 y Win64
 - Cygwin → Ejecutables POSIX y LSB





2.4 Maquinas virtuales

- **Máquinas virtuales a nivel de sistema operativo (Hipervisores):**
 - Son máquinas virtuales que se montan directamente sobre el hardware.
 - Permite instalar varios sistemas operativos de forma paralela.
 - Se establecen así diferentes compartimentos que actúan independientemente.
 - Este tipo de virtualización se utiliza para SO Multiusuarios (servidores), cuando tienen que proveer servicios a sistemas diferentes.
 - El usuario percibe que en realidad se están ejecutando varios sistemas operativos en sus respectivas máquinas.
 - Un ejemplo de maquina virtual Hipervisora: VMware ESX.



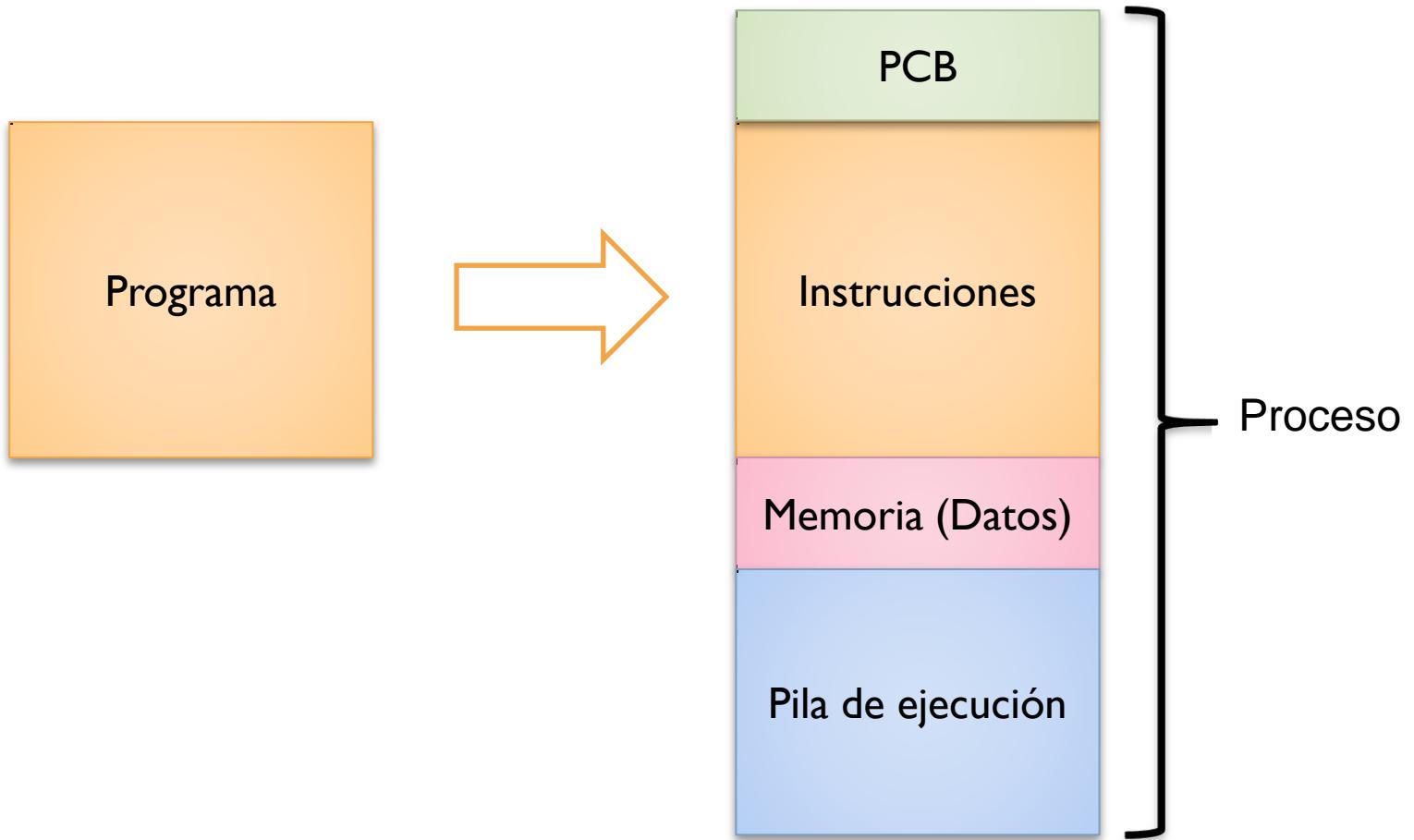


2.5 Gestión de procesos

- Los **procesos** son la instancia en ejecución de un programa.
- Los procesos están compuestos por:
 - La lista ordenada de instrucciones (programa).
 - Los recursos de memoria asignados para su ejecución.
 - Memoria de datos (Memoria Estática) → Variables.
 - Pila de ejecución (Memoria Dinámica)
 - **PCB (Bloque de Control de Proceso):** Guarda el contexto del proceso en ejecución:
 - **Identificador del proceso (PID)**
 - Contenido de los registros del procesador.
 - Ubicación en memoria.
 - Archivos y otros recursos en uso.
 - Prioridad de ejecución.
 - Directorio de trabajo.
 - Privilegios.



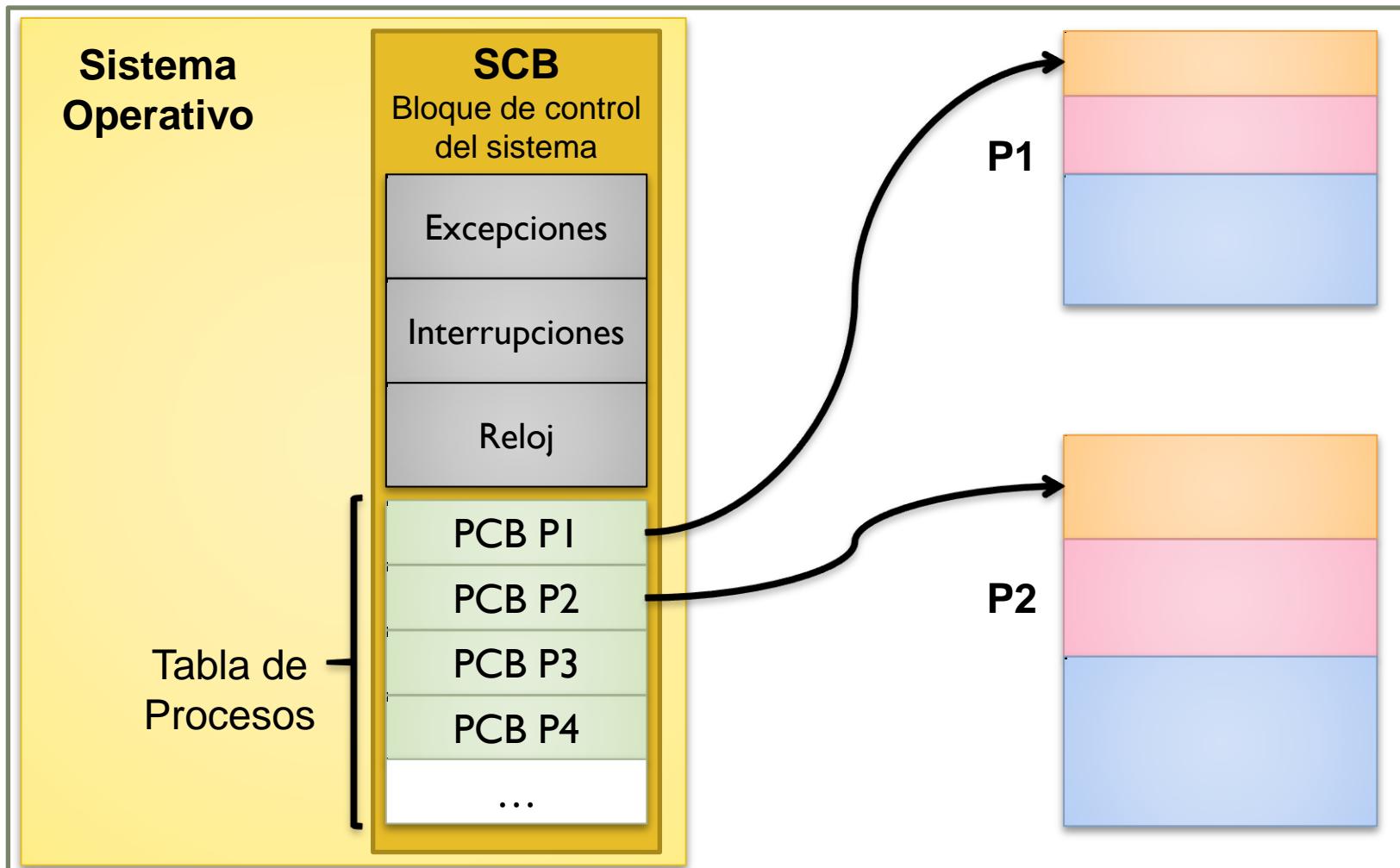
2.5 Gestión de procesos





2.5 Gestión de procesos

RAM





2.5 Gestión de procesos

• **Contexto de un proceso**

- Es la situación de los registros de la CPU de un proceso tomando en consideración un momento en concreto.
- Establecer y controlar el contexto de un proceso es fundamental para un sistema multiprogramado:
 - Cuando se interrumpe un proceso para ejecutar otro, se cede el control al SO, salvando su estado en el PCB.
 - Se pasa de **modo usuario** a **modo kernel**.
 - Se introduce el estado del nuevo proceso en la CPU y se reanuda su ejecución.
 - Se vuelve al **modo usuario**.

Este procedimiento se denomina **cambio de contexto**.



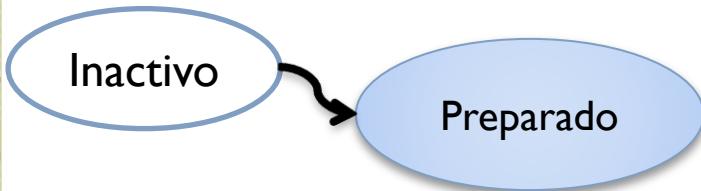
2.5 Gestión de procesos

Inactivo

- **Estado de un proceso**
 - Estado **inactivo**:
 - El proceso se acaba de crear, tiene los recursos de memoria asignados y se ha registrado en la tabla de procesos.
 - Aun no se le ha asignado ninguna prioridad.



2.5 Gestión de procesos



- **Estado de un proceso**
 - Estado **preparado**:
 - El proceso se ha puesto en cola y está listo para ejecutarse, esperará su turno en la **cola de procesos preparados**.



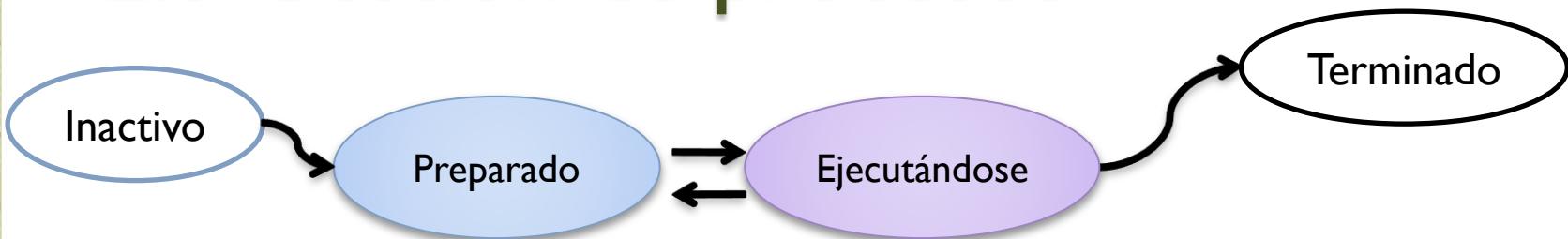
2.5 Gestión de procesos



- **Estado de un proceso**
 - Estado **ejecutándose**:
 - El proceso toma el control de la CPU, se ha cargado desde la **cola de preparados**.
 - El contexto almacenado en el PCB se carga en los registros.
 - Cuando la ejecución cesa, se carga un nuevo proceso desde la **cola de preparados**.



2.5 Gestión de procesos



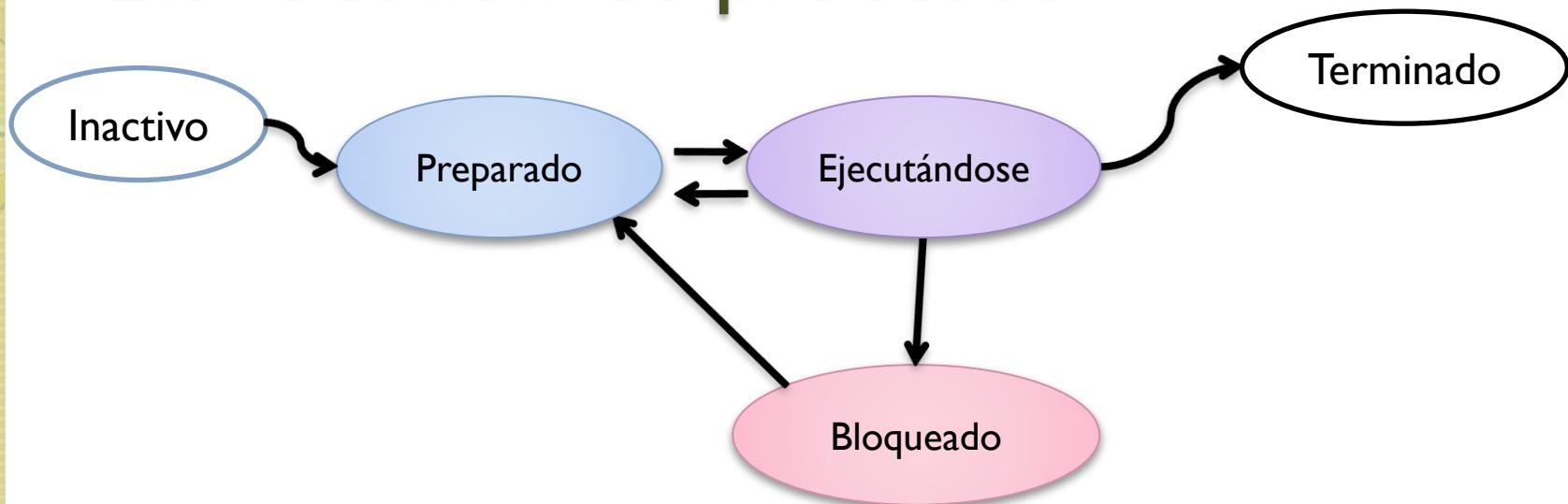
- **Estado de un proceso**

- **Estado Terminado:**

- El proceso ha finalizado no se ejecutará más, se liberan todos los recursos asignados y se borra su PCB.
 - Formas de terminar un proceso:
 - Por solicitud interna o del usuario.
 - Cuando ocurre un error en el proceso.
 - Cuando se le *mata*.



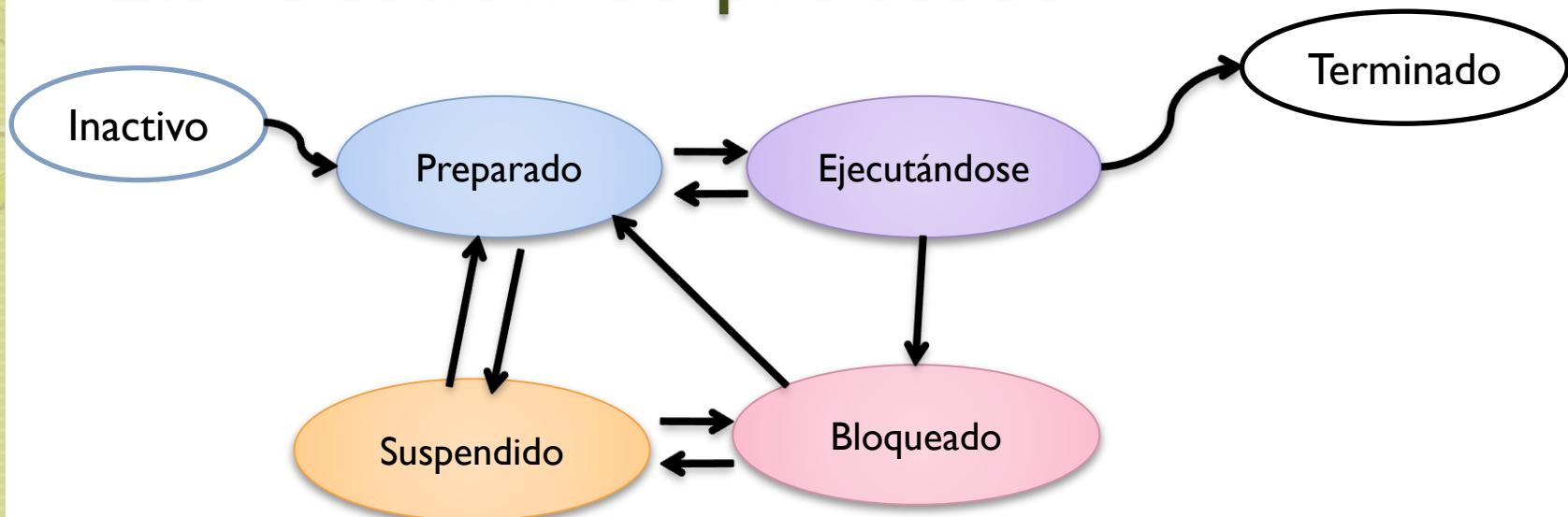
2.5 Gestión de procesos



- **Estado de un proceso**
 - Estado **Bloqueado**:
 - Un proceso que necesita un recurso o solicita una acción que no está disponible en ese momento.
 - El proceso pasa a una de **cola de procesos bloqueados**.
 - Cuando si se ha obtenido una respuesta, el proceso vuelve a la **cola de procesos preparados**.



2.5 Gestión de procesos



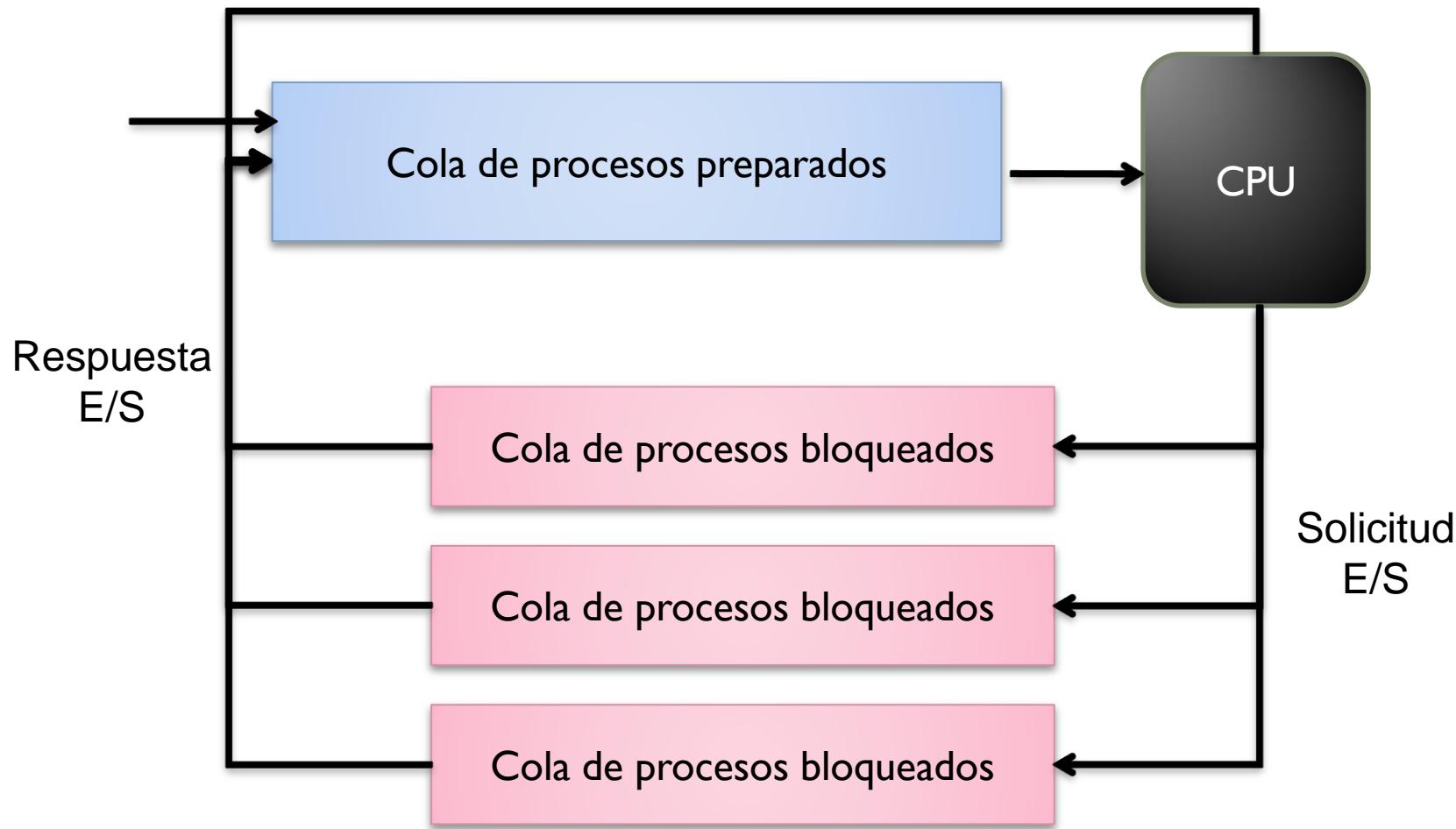
- **Estado de un proceso**

- **Estado Suspendido:**

- En este estado se hallan los procesos que no pueden competir por la CPU, por lo menos de momento.
 - Este proceso normalmente pasa a la **memoria virtual**.



2.5 Gestión de procesos





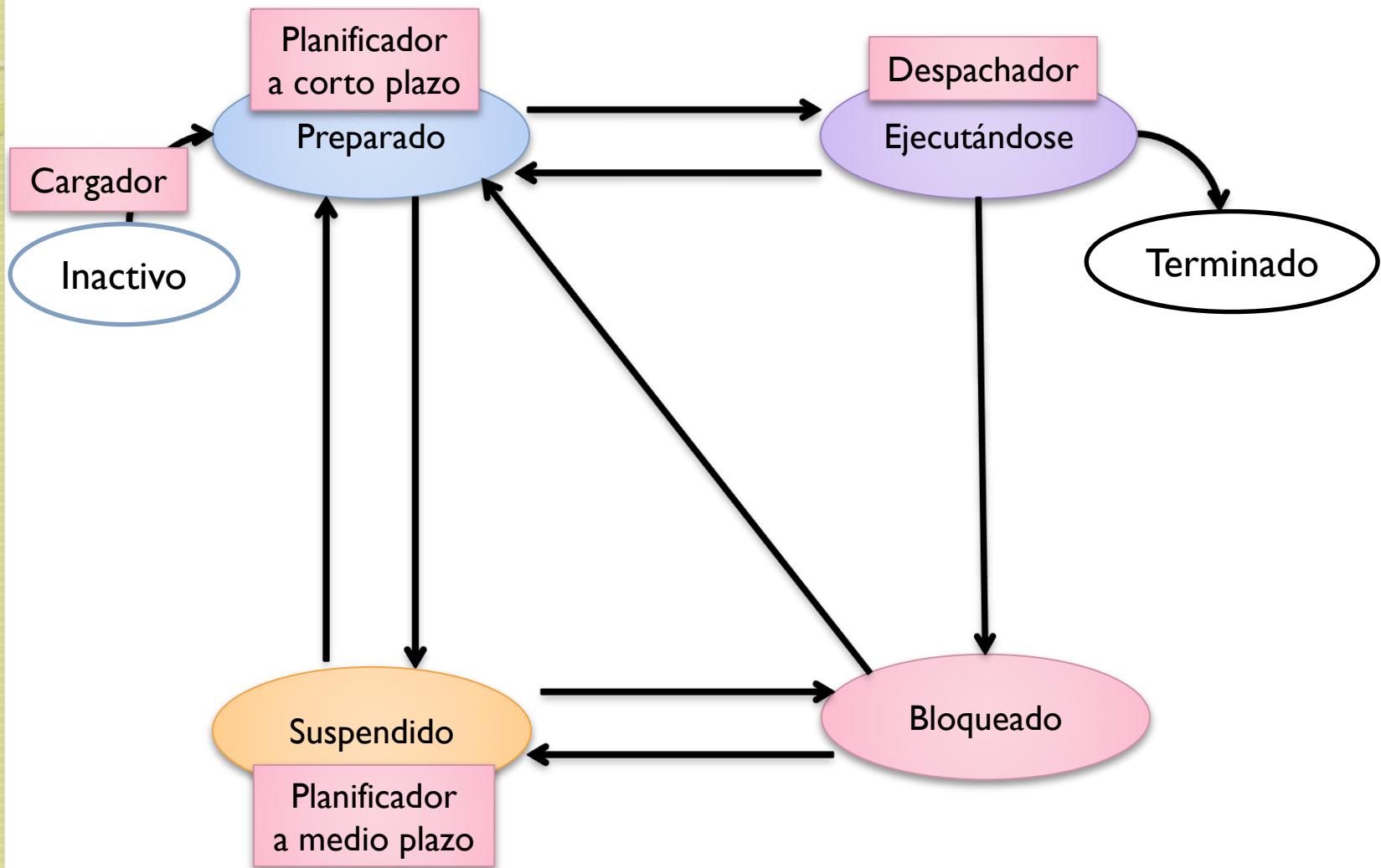
2.5 Gestión de procesos

- **Módulos Implicados**

- **Cargador:** Se encarga de preparar el programa para ser lanzado.
 - Crea el PCB y se le asigna una prioridad.
 - Inserta el proceso en la tabla de procesos.
- **Planificador a corto plazo:** Se encarga de decidir el orden de ejecución de los procesos en la cola de preparados . Se ejecuta unas 50 veces por segundo.
- **Planificador a medio plazo:** Se encarga de decidir que procesos pueden competir por la CPU.
- **Despachador:** Realiza la sustitución del proceso en ejecución por el primero en la cola de preparados o sea. Se ejecuta en cuatro momentos:
 1. Cuando el proceso en ejecución agota su tiempo de CPU.
 2. Cuando el proceso en ejecución es interrumpido por una solicitud de un recurso o acción.
 3. Cuando el proceso de ejecución realiza una llamada al sistema.
 4. Cuando el proceso en ejecución termina.



2.5 Gestión de procesos





2.5 Gestión de procesos

- **Interrupciones**
 - Son señales recibidas por el procesador durante la ejecución de un proceso.
 - Detiene la ejecución actual y devuelve el control al sistema operativo, el cual después de atender la interrupción devolverá la CPU a los procesos.
 - Las interrupciones hacen posible la interactividad en un ordenador.
 - Éstas se gestionan a través del **manejador de interrupciones** y el SCB.
 - Tipos:
 - **Interrupciones Hardware**
 - **Interrupción de la CPU:** Interrupción **síncrona** que indica que el tiempo de CPU se ha agotado para el proceso en ejecución. Hace posible la multiprogramación.
 - **Interrupción de respuesta E/S:** Interrupción **asíncrona** producida por un periférico cuando da una respuesta a una solicitud realizada por un proceso en un momento anterior.
 - **Interrupciones Software (Traps)**
 - La produce una llamada al sistema de una aplicación o usuario.
 - Se produce de forma **síncrona** con la ejecución, producen el estado bloqueado en el proceso.



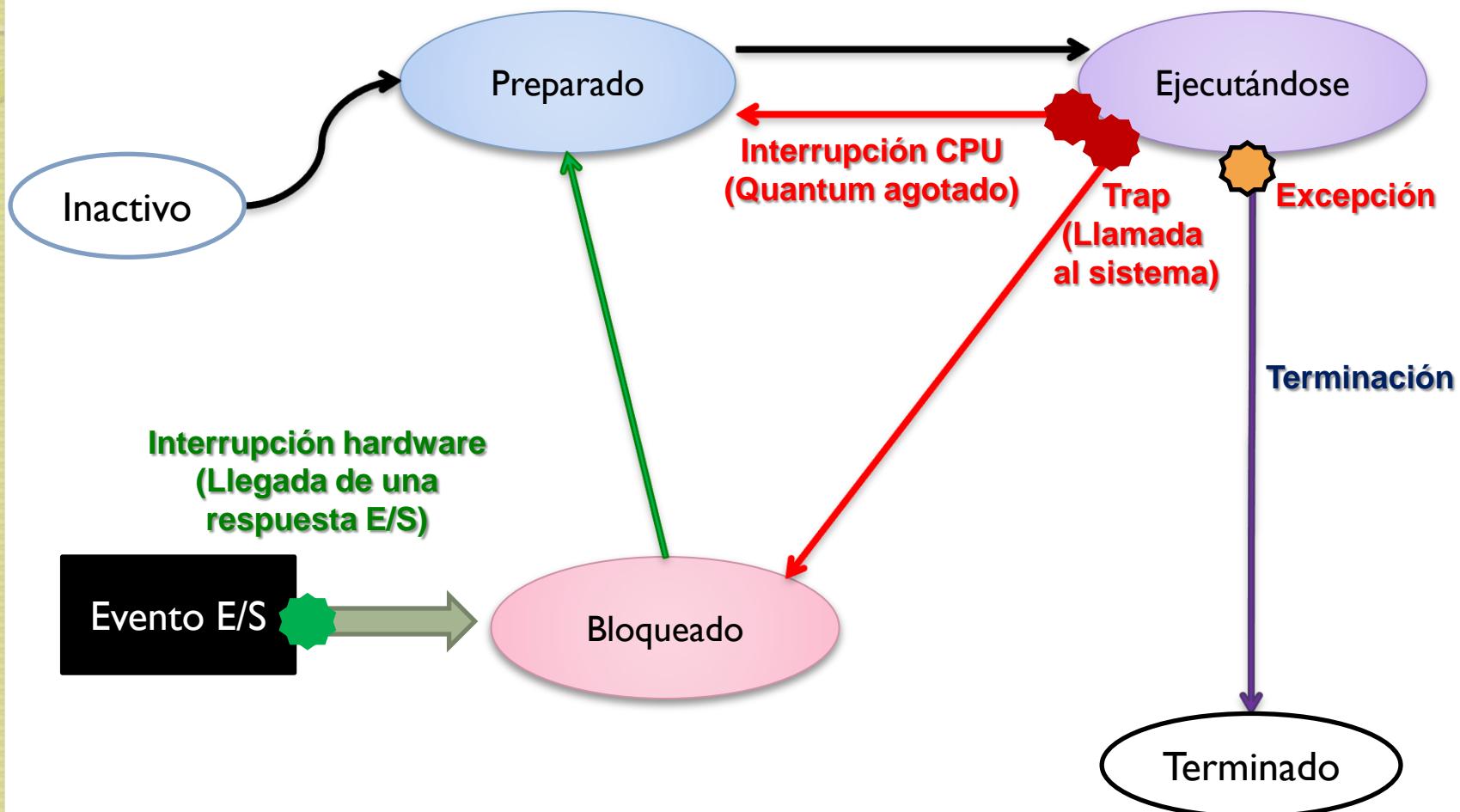
2.5 Gestión de procesos

- **Excepciones**

- La produce una situación anormal en la CPU producida por una instrucción errónea (división por cero, fallo de dirección...). La interrupción es generada por la propia CPU.
- Se produce de forma **síncrona** con la ejecución.
- En este caso se llama al gestor de excepciones que junto al SCB tratan de resolver el problema.
- Pueden pasar tres cosas:
 - Que el error sea **fatal**: afectan al sistema y este puede incluso bloquearse.
 - Que el error sea **crítico**: no afectan al SO, sólo al proceso el cual no puede continuar. El SO normalmente decide matar al proceso.
 - Que el error sea **recuperable**: realizando algunos ajustes, el proceso puede continuar, aunque quizás no lo haga correctamente.



2.5 Gestión de procesos

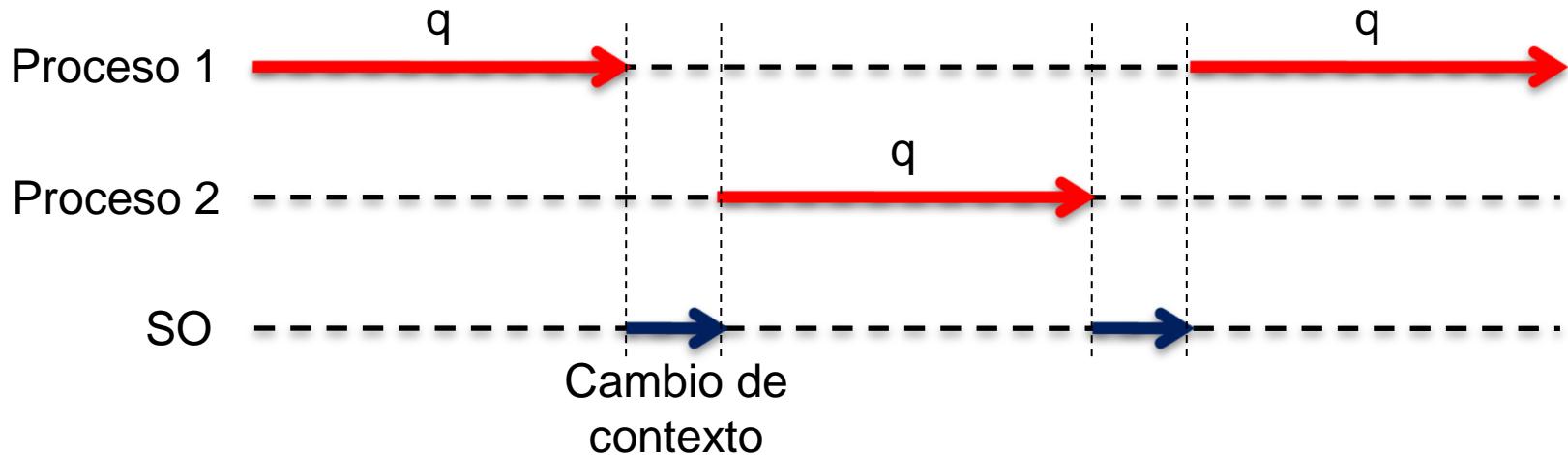




2.5 Gestión de procesos

- **El Despachador**

- Cuando un proceso está ejecutándose, las variables que en ese momento se están utilizando, están cargadas en los registros de la CPU.
- Cuando el *quantum* de tiempo se agota o se produce una interrupción. El **despachador** se activa y detiene proceso, salva los valores actuales de estos registros en el PCB de ese proceso.
- El despachador también se encarga de cargar los datos del nuevo proceso en los registros de la CPU.
- Los sistemas de multiprogramados realizan de 100 a 1000 cambios de contexto por segundo.





2.5 Gestión de procesos

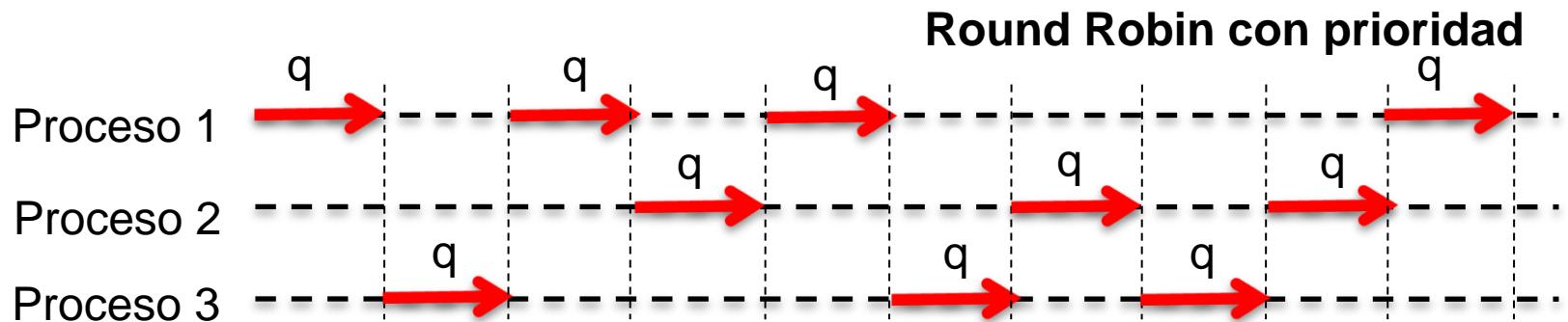
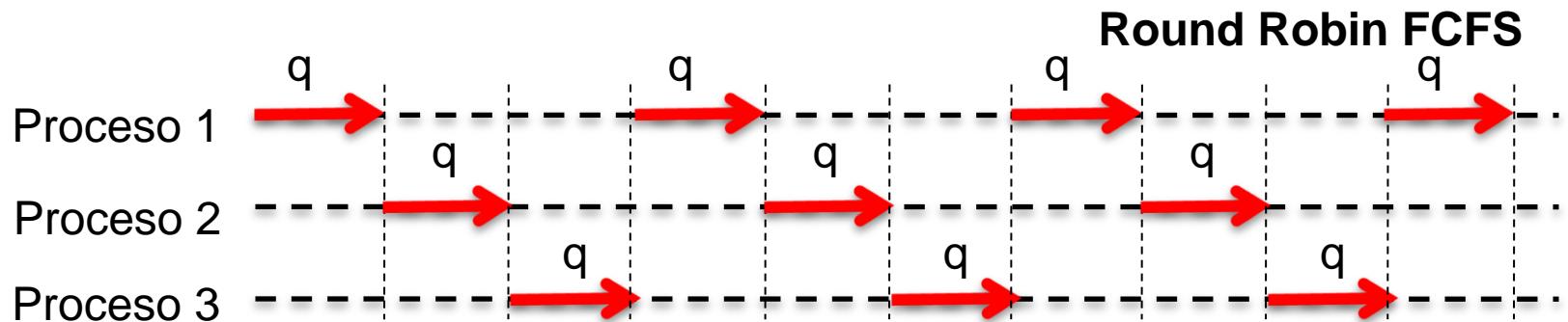
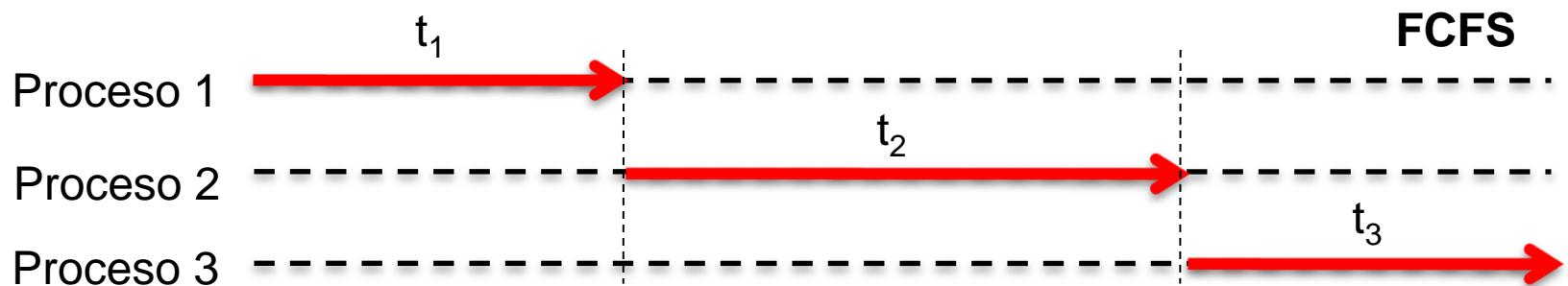
• El Planificador a Corto Plazo

- Cuando se activa (lo hace varias veces por segundo) reorganiza el orden de los procesos en la **cola de preparados**.
- Dependiendo de la complejidad del SO, los algoritmos ordenación que utiliza en planificador a corto plazo se pueden clasificar en:
 - **No apropiativas:** Una vez que se le asigna la CPU a un proceso, no se le retira hasta que éste finalice o se bloquee. Esto es común en el proceso por lotes.
 - FCFS (First Come First Served)
 - SJF (Shortest Job First)
 - **Apropiativas:** Cuando el sistema operativo puede apropiarse de la CPU cuando se consideré oportuno. Necesario en la **Multiprogramación**.
 - Round-Robin FCFS
 - **Round-Robin con prioridad**



2.5 Gestión de procesos

- **El Planificador a Corto Plazo**





2.5 Gestión de procesos

- **Sistema de prioridades**

- En el momento en el que es creado un proceso, se le asigna un número que representa su prioridad base.
- Muchas veces este número irá cambiando dinámicamente debido dependiendo del entorno y del tiempo que ha estado en ejecución el proceso.
- En los sistemas **Windows NT** vamos a tener prioridades del 0 al 31:
 - **0** → **proceso suspendido**
 - **1-15** → **aplicaciones de uso compartido** (dinámicas)
 - 4 → “Debajo de lo normal”
 - 7 → “Normal”
 - 9 → “Mas de lo normal”
 - 13 → “Alta”
 - **16-31** → **aplicaciones de tiempo real** (fijas)
 - 24 → “Tiempo real”
- En los sistemas **tipo Unix** vamos a tener prioridades del 0 al 159:
 - **0 - 59** → **aplicaciones de uso compartido** (dinámicas)
 - **60-99** → **aplicaciones kernel** (procesos bloqueados)
 - **100-159** → **aplicaciones de tiempo real** (fijas)

En estos sistemas, el valor *nice* le indica al SO entre qué valores puede cambiar la prioridad dinámicamente. Por defecto este valor es cero, pero se puede cambiar entre -20 y 19 puntos de prioridad.



2.5 Gestión de procesos

- **Grado de multiprogramación**

- Es el número de procesos que se están ejecutando en relación al quantum de tiempo.
- Por lo general, el uso de la CPU aumenta con el grado de multiprogramación.
- La multiprogramación por lo general implica un potencial desaprovechamiento de los recursos de sistema.
- Un sistema con un GM bajo se comporta como un sistema monoprogramado. Pero el aprovechamiento de los recursos del sistema es mayor.
- Para aumentar el GM debemos reducir la duración el quantum de tiempo.
- Si en *quantum* es demasiado bajo, el SO se ejecutará en total más tiempo que los procesos de usuario.



2.5 Gestión de procesos

- **Técnicas de implementación de procesos**
 - **Bifurcación**
 - Cuando un proceso lo requiera, este puede efectuar una llamada al sistema para solicitar una instancia de si mismo.
 - Los dos procesos pasan a tener una relación de dependencia **Padre-Hijo**.
 - Esto se efectúa para realizar una tarea en paralelo que puede estar relacionada o no con la del padre.
 - El hijo se crea como una copia idéntica de su padre, sólo se diferencia en su PID.
 - Durante la ejecución el padre y el hijo divergirán, pero conservaran su dependencia.
 - En los SO que se utiliza esta técnica todos los procesos son descendientes de un único proceso.
 - **Sistemas Windows NT** → system.exe (PID 4)
 - **Sistemas basados Unix** → init (PID 1)

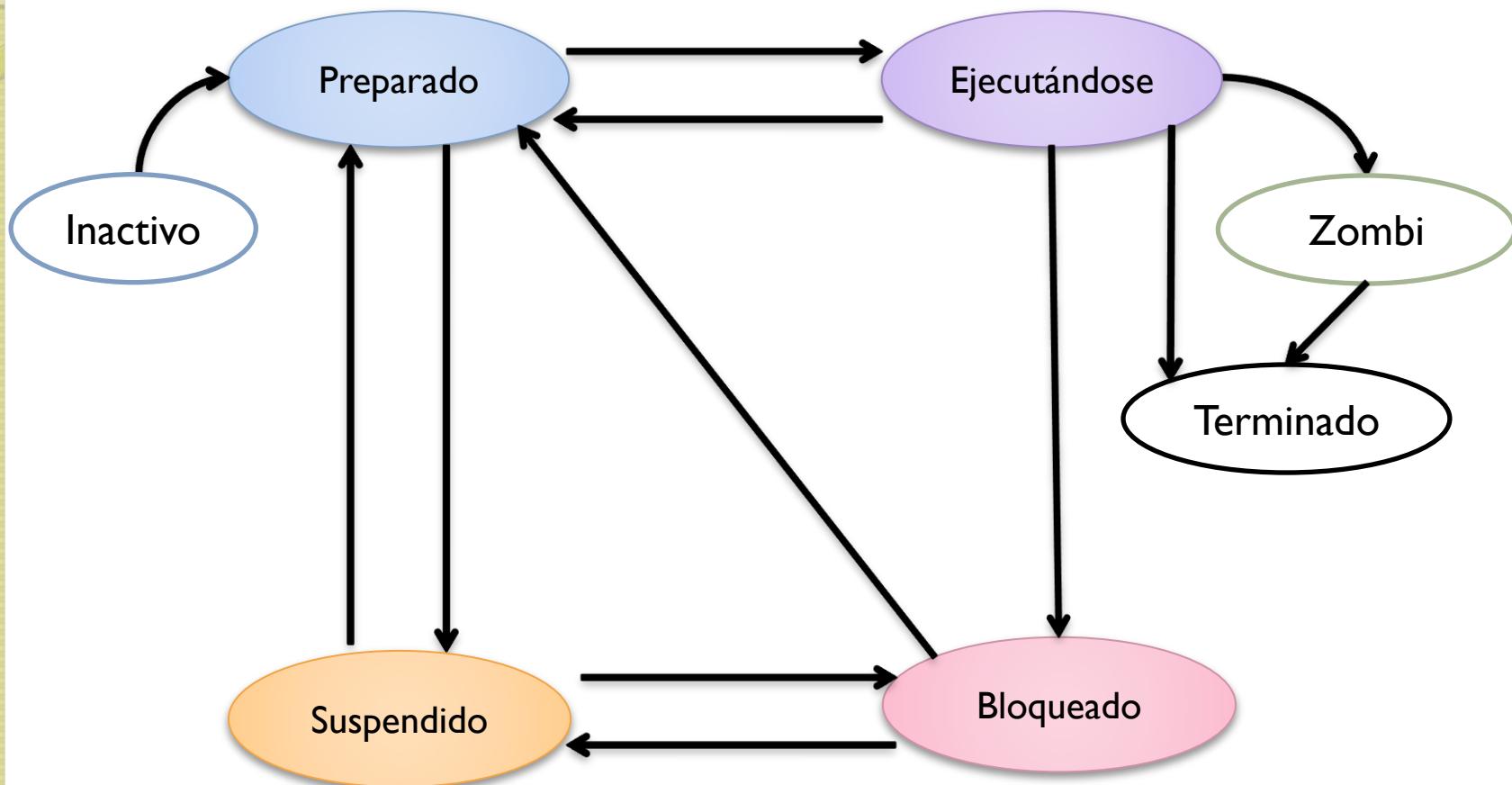


2.5 Gestión de procesos

- **Técnicas de implementación de procesos**
 - **Bifurcación**
 - Normalmente el proceso padre se bifurca para obtener algún servicio del hijo.
 - Si hijo termina y tiene datos relevantes a la ejecución de su padre, lo normal es que hasta que éste último termine, el hijo espere en estado **Zombi**.
 - Si padre es el que termina antes que su hijo, para no convertirse en **Zombi** otro proceso pasara a ser su padre.
 - **Servicio de sistema Windows NT** → services.exe
 - **Proceso de usuario Windows NT** → explorer.exe
 - **Proceso tipo Unix** → init
 - Si un padre es matado o se cierra anormalmente, todos sus hijos mueren con él.

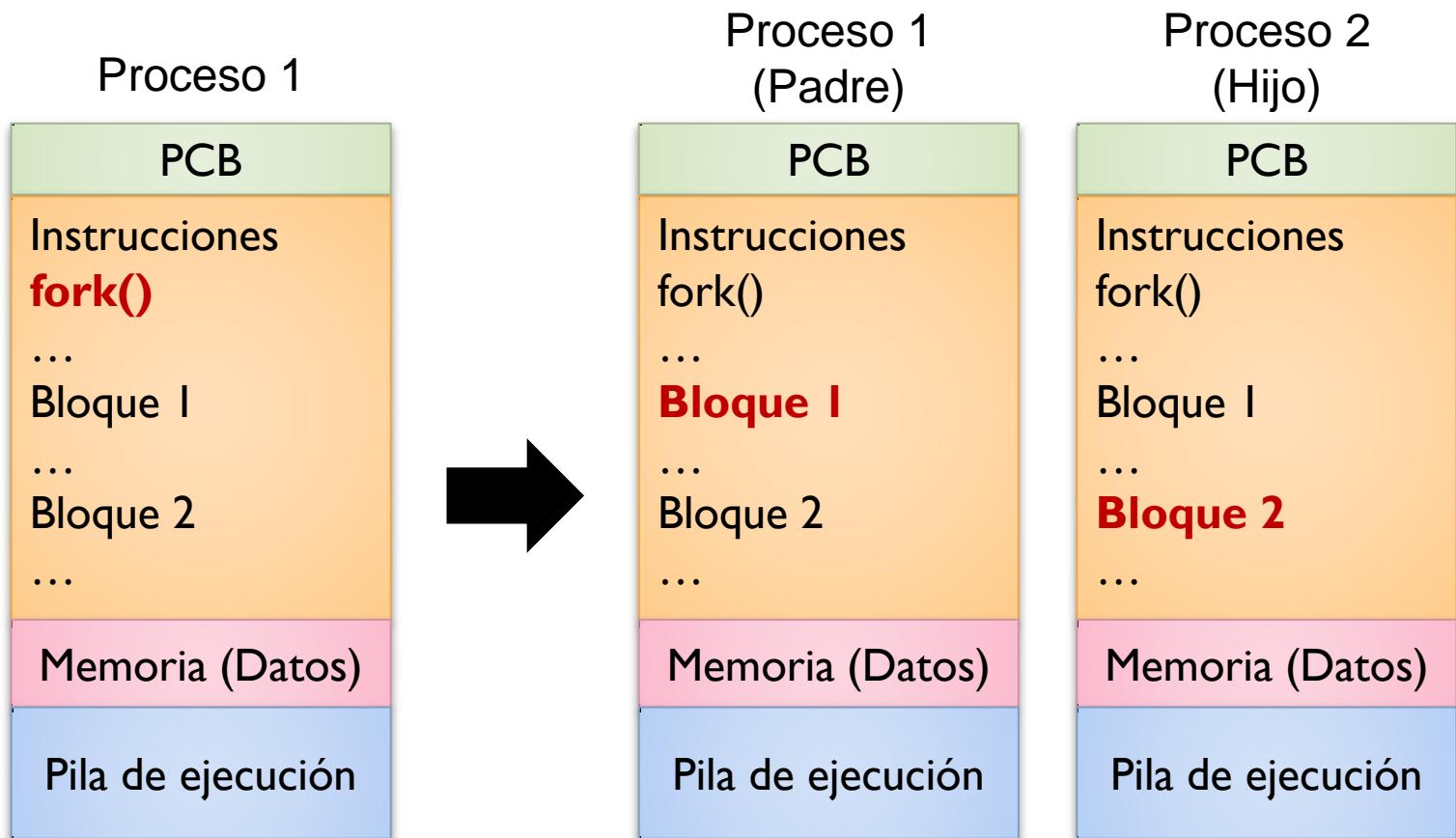


2.5 Gestión de procesos





2.5 Gestión de procesos





2.5 Gestión de procesos

- **Técnicas de implementación de procesos**
 - **Bifurcación**
 - En LSB `fork()` devuelve el PID del proceso hijo en caso de tratarse del padre y en caso de tratarse del hijo, devuelve cero.

```
switch (fork())
{
    case -1:
        Código en caso de error
        break;
    case 0:
        Código del proceso hijo
        break;
    default:
        Código del proceso padre
}
```



2.5 Gestión de procesos

- **Técnicas de implementación de procesos**
 - **Hilo de ejecución**
 - El proceso principal posee una serie de recursos asignados y una serie de **hilos** o **hebras** (*thread*) que lo forman.
 - Cada hilo constituye ahora una unidad de ejecución que se ejecuta concurrentemente con los demás hilos del proceso madre.
 - Durante la ejecución del proceso, cada hilo realiza una tarea diferente pero orientada a un mismo fin.
 - Todos los hilos comparten los recursos de memoria.
 - El uso de hilos mejora el rendimiento ya que:
 - La creación, manejo y terminación de un hilo es más rápido que la de un proceso tradicional.
 - Los hilos se comunican entre más fácilmente que los procesos tradicionales (entre hilos no hay protección de memoria).
 - La ejecución paralela evita retrasos debido a operaciones de E/S.



2.5 Gestión de procesos

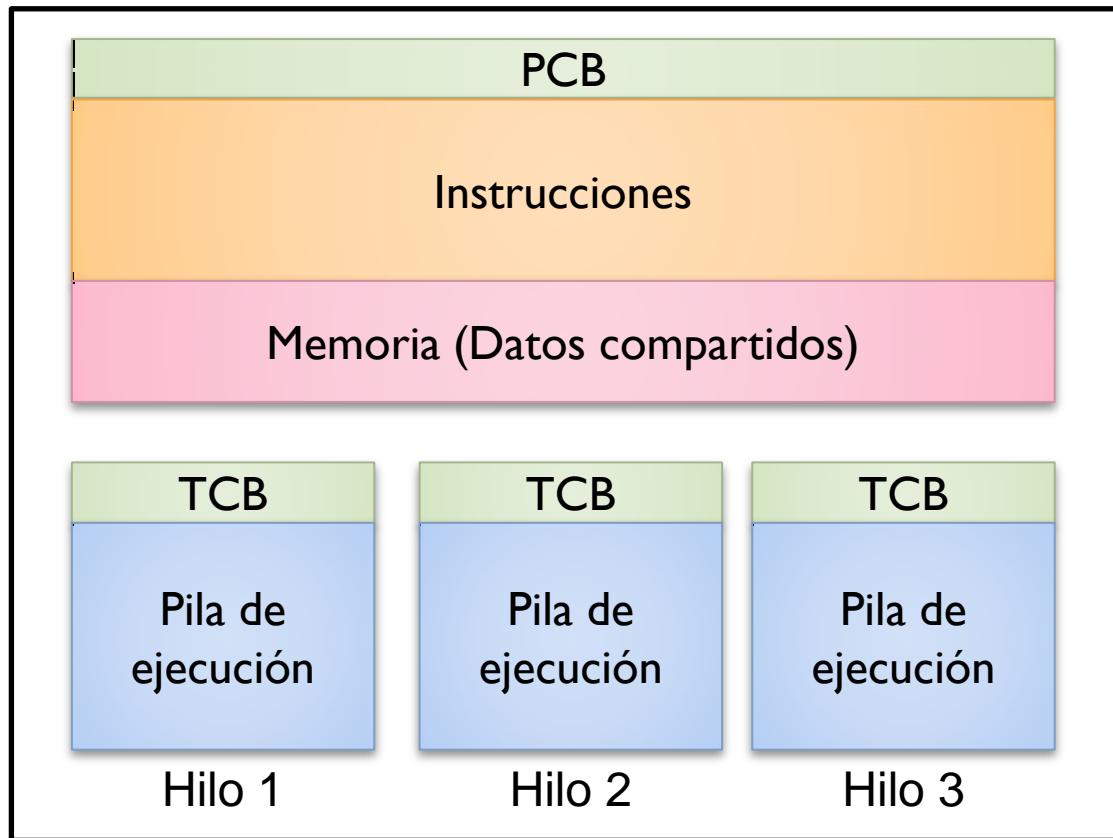
- **Técnicas de implementación de procesos**
 - **Hilo de ejecución**
 - Estado del proceso madre:
 - Si algún hilo está en **ejecución**, el proceso madre estará en **ejecución**.
 - Si ningún hilo de un proceso está en **ejecución**, pero hay alguno en estado **preparado**, el proceso madre esta en estado **preparado**.
 - Si todos los hilos se **bloquean**, el proceso madre pasa al estado **bloqueado**.
 - Si el SO decide **suspender** un proceso, **suspende** todos sus hilos con él.
 - Un proceso se **termina** cuando terminan todos sus hilos.



2.5 Gestión de procesos

- **Técnicas de implementación de procesos**
 - **Hilo de ejecución**

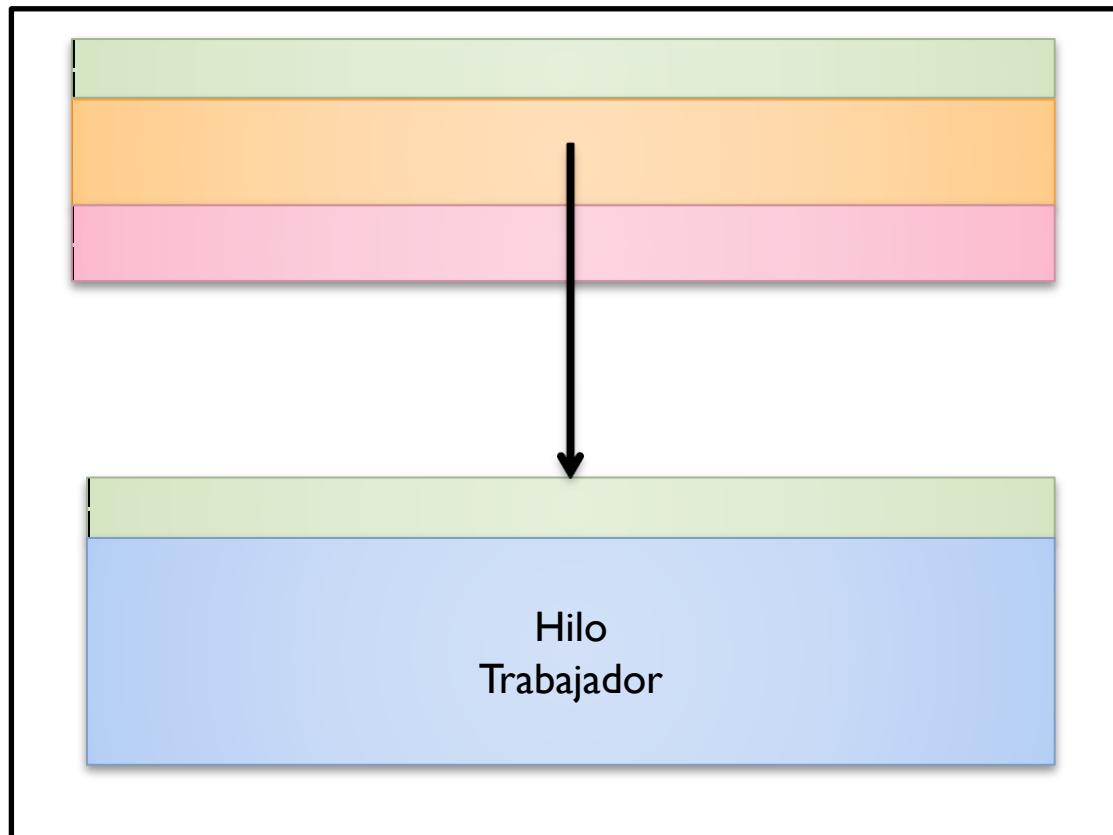
Proceso Madre





2.5 Gestión de procesos

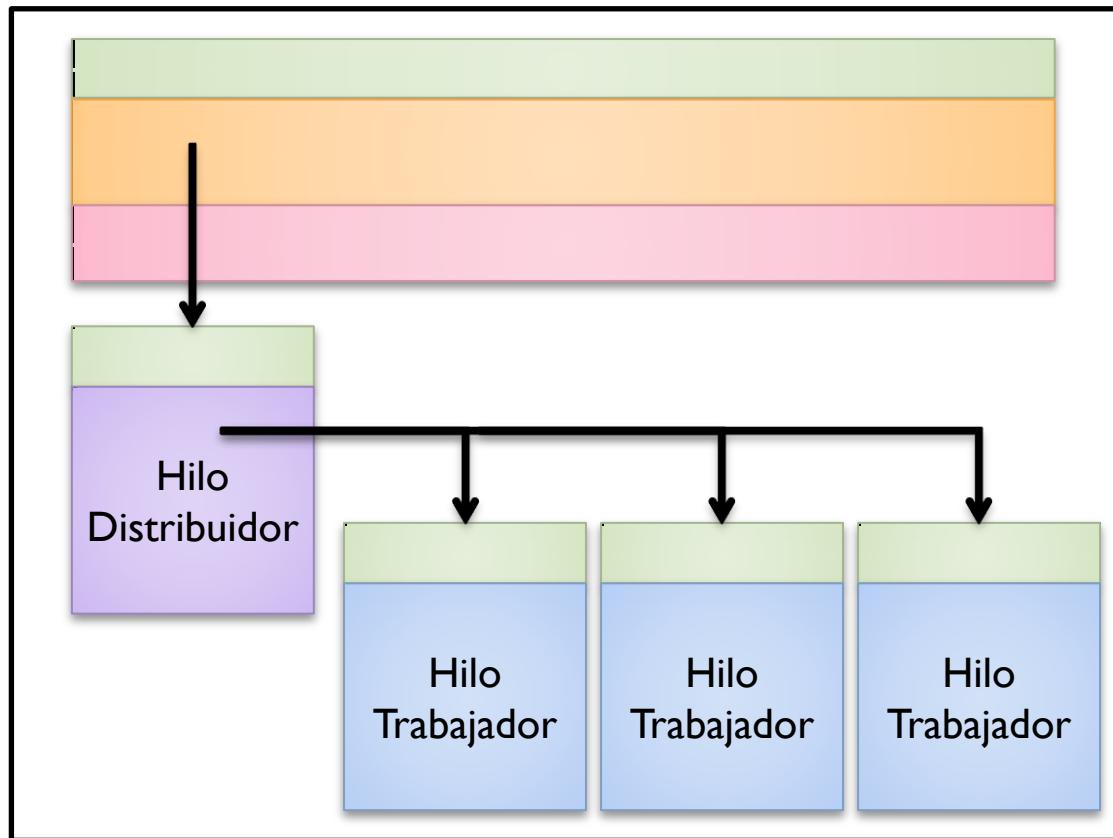
- **Técnicas de implementación de procesos**
 - **Hilo de ejecución**





2.5 Gestión de procesos

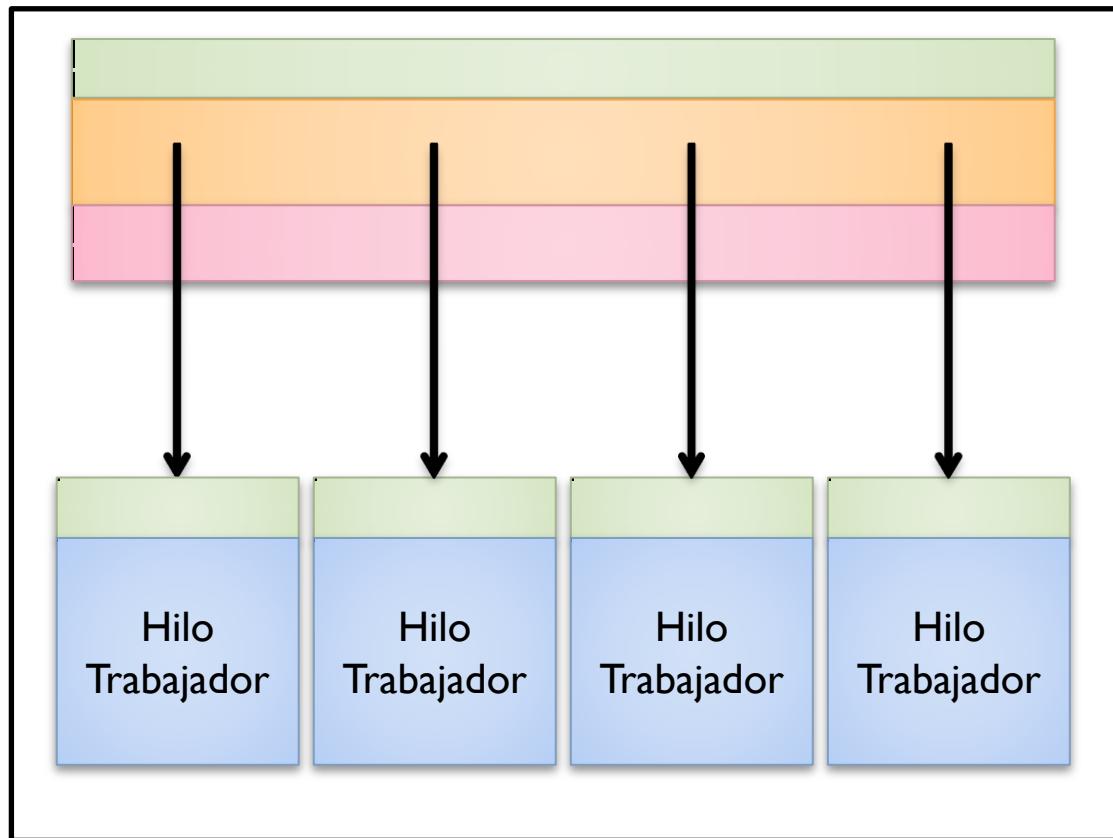
- **Técnicas de implementación de procesos**
 - **Hilo de ejecución**





2.5 Gestión de procesos

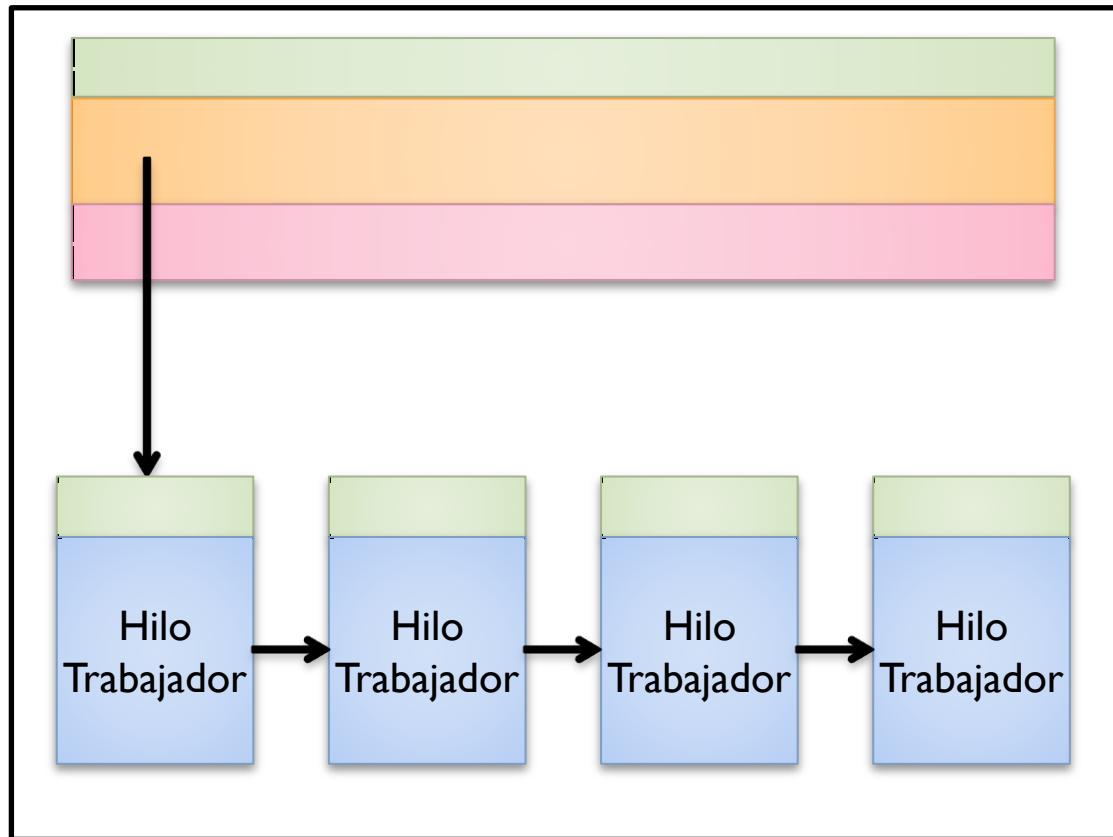
- **Técnicas de implementación de procesos**
 - **Hilo de ejecución**





2.5 Gestión de procesos

- **Técnicas de implementación de procesos**
 - **Hilo de ejecución**





2.5 Gestión de procesos

- **Técnicas de implementación de procesos**
 - **Hilo de ejecución**
 - En lenguaje C, la biblioteca con las funciones de creación de hilos están en **pthread.h**
 - En Java, podemos manipular hilos con la clase **java.lang.Thread**
 - Crear hilos →
 - Asignar prioridad →
 - Iniciar el hilo →
 - Detener el hilo →
 - Reanudar hilos →
 - Consultar si vive →
 - Obtener etiqueta →
 - Asignar etiqueta →
- public void run()**
public void setPriority(int P)
public void start() (llama a run())
public void suspend()
public void resume()
public boolean isAlive()
public String getName()
public void setName(String etiq)

```
class HiloEjem extends Thread {  
    public HiloEjem(String str) {  
        super(str);  
    }  
    public void run(){  
        for (int i = 0; i < 10; i++) {  
            System.out.println(i + " " + getName());  
            try {  
                sleep((int)(Math.random() * 1000));  
            } catch (InterruptedException e) {}  
        }  
        System.out.println("Final:" + getName());  
    }  
}  
class Prueba {  
    public static void main(String[] args) {  
        new HiloEjem("Marco").start();  
        new HiloEjem("Polo").start();  
    }  
}
```

0 Marco
0 Polo
1 Polo
1 Marco
2 Marco
2 Polo
3 Polo
3 Marco
4 Marco
4 Polo
5 Marco
5 Polo
6 Polo
6 Marco
7 Marco
7 Polo
8 Polo
9 Polo
8 Marco
Final: Polo
9 Marco
Final: Marco



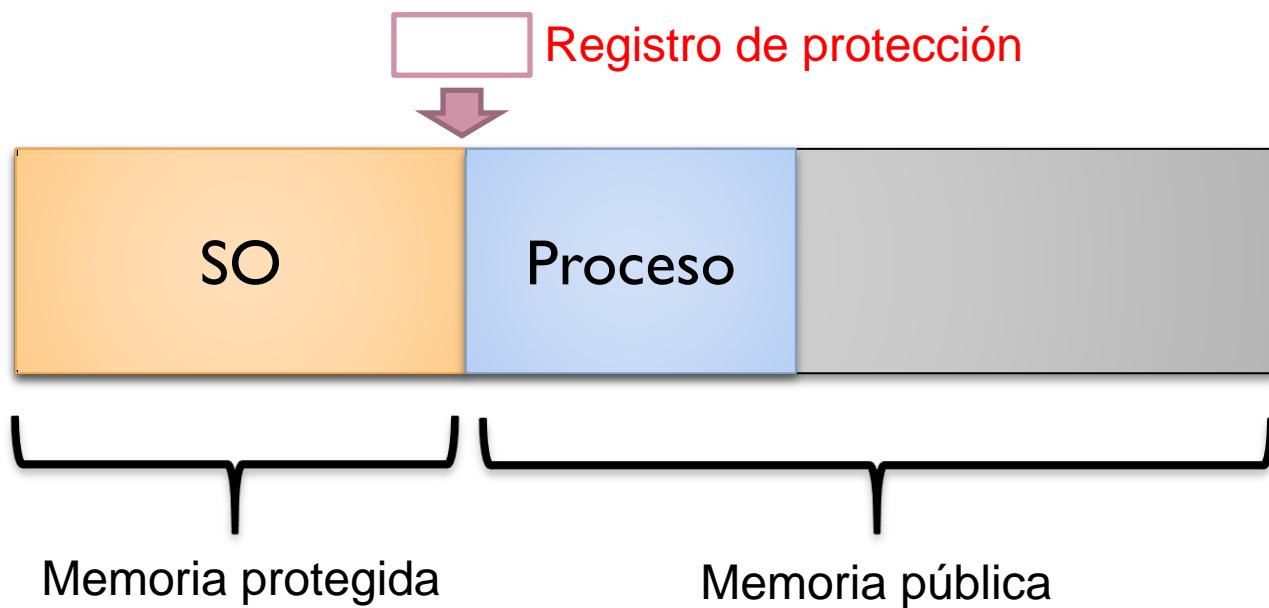
2.6 Gestión de memoria

- Esta gestión la realiza el **administrador de memoria**, este módulo del núcleo tiene los siguientes cometidos:
 - Guarda las direcciones de memoria de las regiones que están en uso y de las que no.
 - Asigna espacio de memoria de los nuevos procesos.
 - Libera la memoria de los procesos terminados.
 - Protección de los espacios de memoria



2.6 Gestión de memoria

- **Gestión en los sistemas monoprogramados**
 - La memoria se dividía en dos regiones.
 - Una región protegida para el sistema operativo.
 - Una región pública para ubicar un único proceso.
 - Se utiliza un registro para guardar la dirección de memoria que divide las dos regiones.

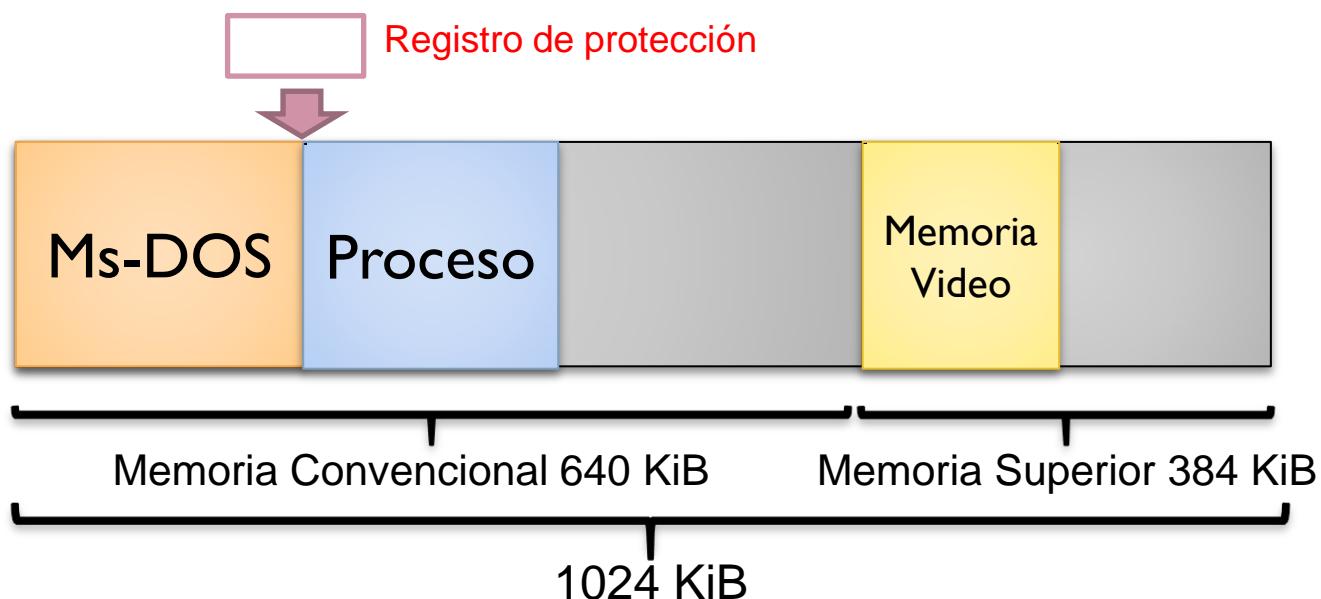




2.6 Gestión de memoria

- **Gestión en memoria en Ms-DOS:**

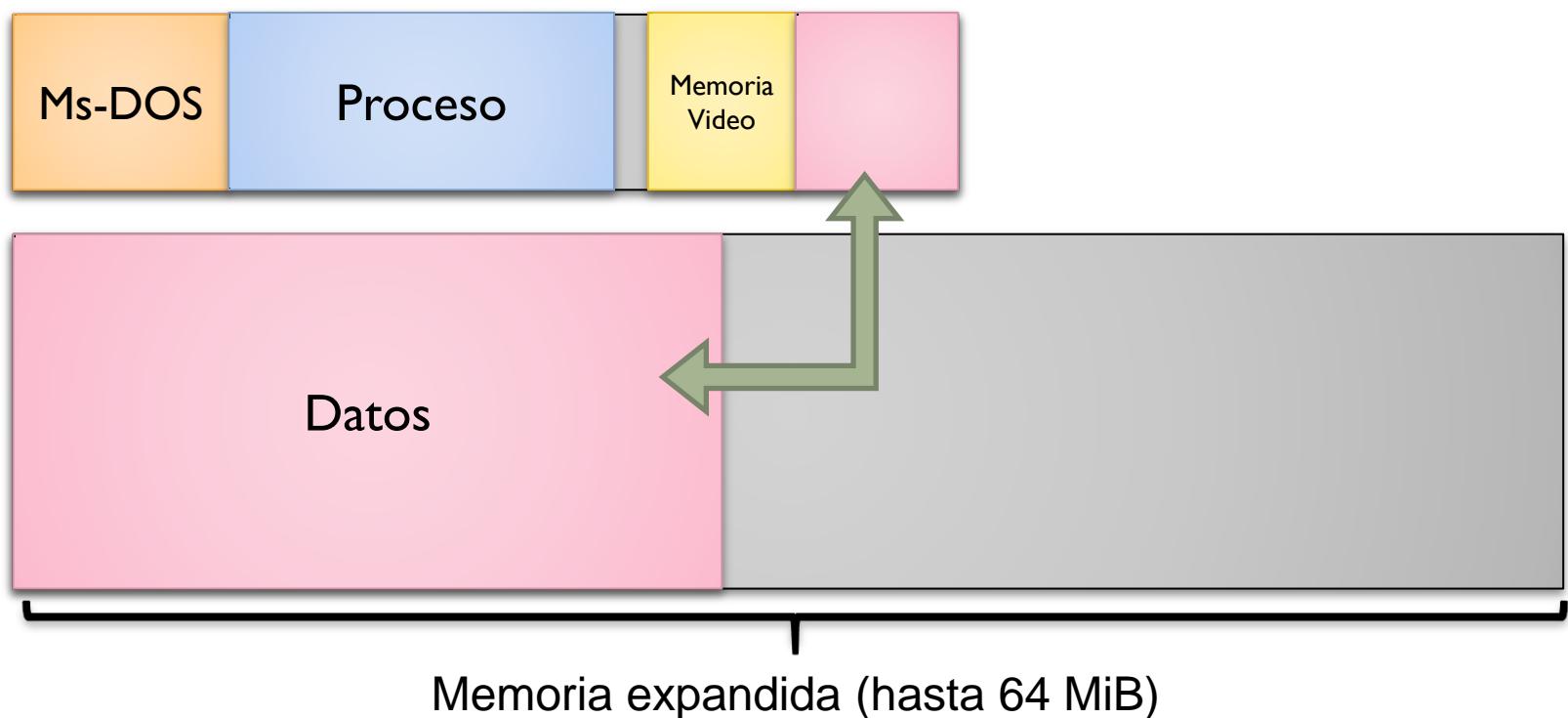
- Ms-DOS solo puede direccionar 2^{20} palabras de 8 bits (1 MiB):
 - Memoria convencional (640 KiB)
 - Núcleo del Ms-DOS, drivers y entorno de texto (comman.com)
 - Aplicaciones.
 - Memoria superior UMB (384 KiB)
 - Memoria de Video (128 KiB)
 - Espacio no utilizado (256 KiB)





2.6 Gestión de memoria

- **Gestión en de memoria en Ms-DOS:**
 - Memoria expandida EMS (hasta 64 MiB)
 - Necesita un driver adicional para poder utilizarla (EMM384.EXE)
 - No se pueden ejecutar instrucciones contenidas en la EMS, antes se deben de transvasar a la memoria de UMB





2.6 Gestión de memoria

- **Gestión en memoria en Ms-DOS:**

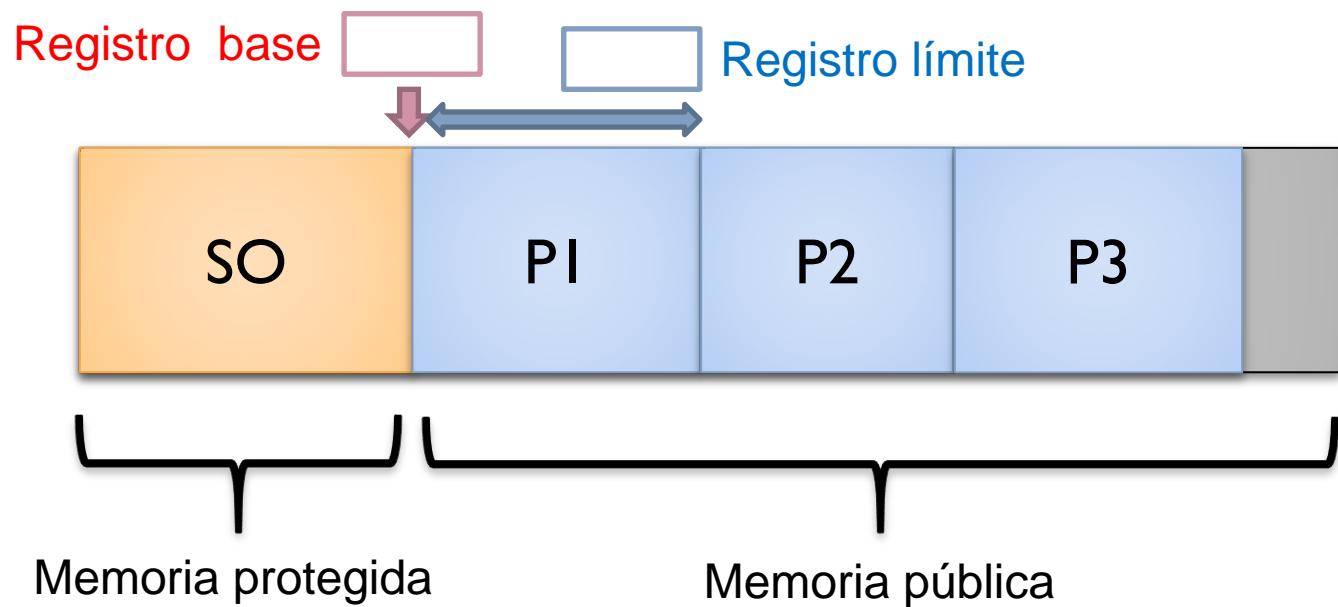
- Memoria extendida XMS (hasta 4 GiB)
 - La evolucion de los procesadores permitieron eventualmente el aprovechamiento de la memoria mas alla de los 1024 KiB
 - Necesita un conjunto de drivers de memoria especiales (RAMDRIVE.SYS)
 - La memoria XMS permite al Ms-DOS funcionar en modo multiprogramado → **Primeras versiones de Windows**





2.6 Gestión de memoria

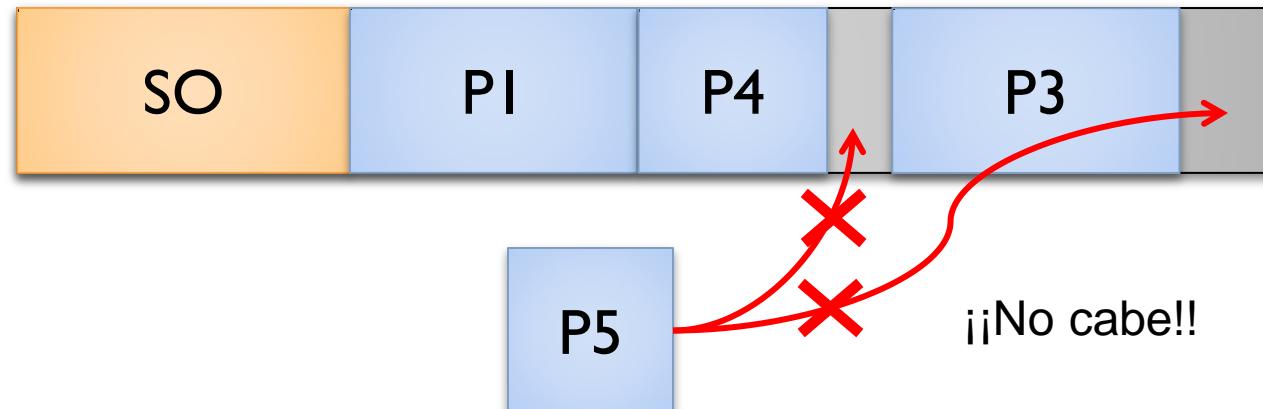
- **Gestión en los sistemas multiprogramados:**
 - La memoria puede contener varios procesos.
 - Se utilizan dos registro de protección para limitar el acceso a memoria:
 - Registro base: marca el inicio de una partición de memoria.
 - Registro límite: indica el tamaño de la partición.
 - Los registros de protección se guardan en el PCB en un cambio de contexto.





2.6 Gestión de memoria

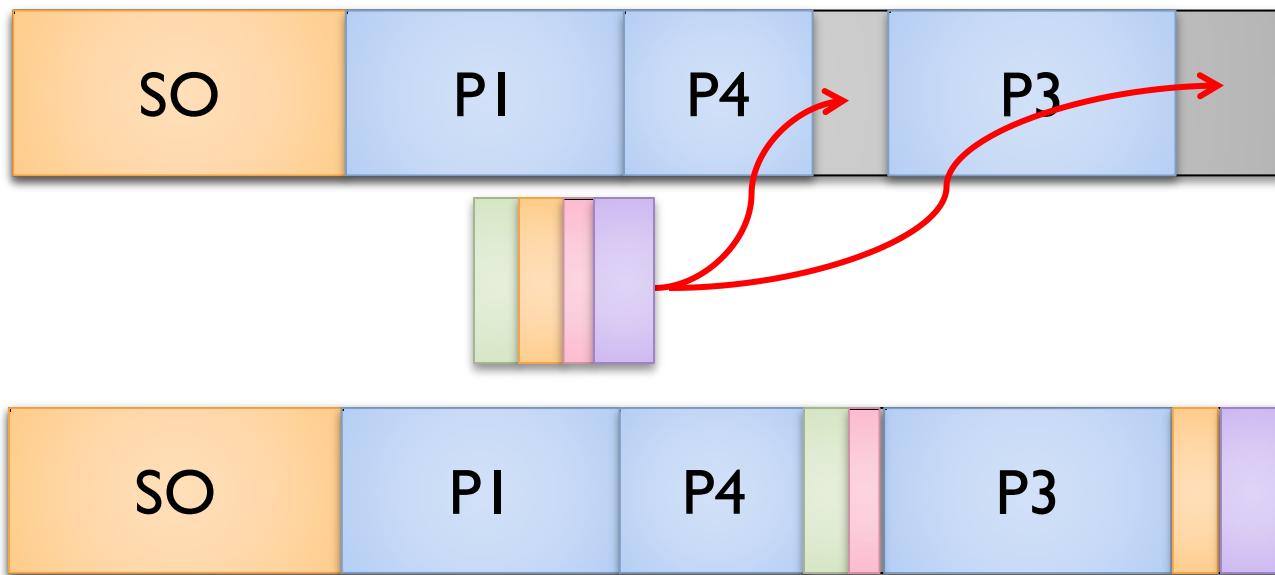
- **Aparición de huecos (Fragmentación externa):**
 - Ocurre cuando se libera memoria de los procesos y los huecos resultantes no son aprovechables





2.6 Gestión de memoria

- **Soluciones a la fragmentación externa:**
 - **Segmentación:**
 - Dividir los procesos en segmentos de sus partes fundamentales:
 - Hebras
 - Instrucciones, memoria de datos, pila...
 - Subrutinas, subespecies de memoria...
 - No lo soluciona del todo, los segmentos originan una fragmentación más pequeña.



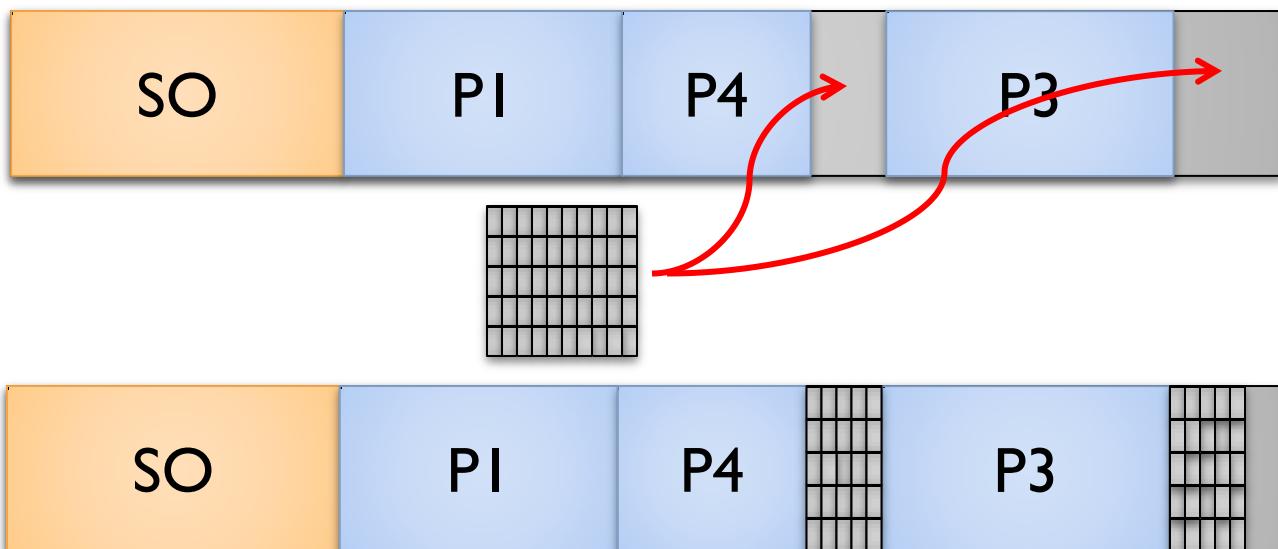


2.6 Gestión de memoria

- Soluciones a la **fragmentación externa**:

- **Paginación:**

- Se dividen los procesos en porciones iguales de minúsculo tamaño llamadas páginas.
 - Las páginas corren completamente el problema de la fragmentación externa.



- En muchos casos, las páginas no se utilizan completamente (la última página del proceso) aparece entonces un nuevo problema: la **fragmentación interna**.



2.6 Gestión de memoria

- Soluciones a la **fragmentación externa**:
 - **Segmentación paginada**:
 - Combina las técnicas de segmentación y paginación.
 - Esta técnica es la que se utiliza en los sistemas operativos actuales.
 - **La compactación** (o desfragmentación de memoria):
 - Es una técnica alternativa que alivia la fragmentación externa en caso de no existir la paginación.
 - Consiste en redistribuir la memoria para eliminar los huecos.
 - Se puede combinar con la segmentación
 - Es muy costosa por lo que no se puede realizar con frecuencia.
- Soluciones a la **fragmentación interna**:
 - Disminuir el tamaño de las páginas:
 - Podemos reducir la fragmentación interna a niveles prácticamente nulos.
 - Cuanto más reducimos el tamaño de página, menor es la eficiencia de la gestión de memoria.
 - Normalmente se toma un valor entre 2 y 16 KiB. Por defecto se va a trabajar con páginas de 4 KiB

Ejercicio

Tenemos una memoria RAM de una capacidad de 2,5 GiB exactamente, dividida en páginas de 16 KiB. En el sistema tenemos los siguientes procesos en ejecución:

SO	12 030 KiB
P1	123 KiB
P2	302 KiB
P3	411 KiB
P4	10 023 KiB

Calcula:

1. Cuantas páginas caben en total en la memoria.
2. Cuantas páginas ocupan el sistema operativo y los procesos.
3. Cuanto se desperdicia en total con la fragmentación interna.
4. Cuanta memoria recuperaríamos si hubiésemos utilizado páginas de 8 KiB.

I. Cuantas páginas caben en total en la memoria.

$$2.5 \text{ GiB} \cdot 1024 \text{ MiB/GiB} \cdot 1024 \text{ KiB/MiB} = 2621440 \text{ KiB}$$

$$2621440 \text{ KiB} / 16 \text{ KiB/pag} = 163840 \text{ pag}$$

2. Cuantas páginas ocupan el sistema operativo y los procesos.

SO: $12030 \text{ KiB} / 16 \text{ KiB/pag} = 751,875 \text{ pag}$ **(752 pag)**

P1: $123 \text{ KiB} / 16 \text{ KiB/pag} = 7,6875 \text{ pag}$ **(8 pag)**

P2: $302 \text{ KiB} / 16 \text{ KiB/pag} = 18,875 \text{ pag}$ **(19 pag)**

P3: $411 \text{ KiB} / 16 \text{ KiB/pag} = 25,6875 \text{ pag}$ **(26 pag)**

P4: $10023 \text{ KiB} / 16 \text{ KiB/pag} = 626,4375 \text{ pag}$ **(627 pag)**

3. Cuanto se desperdicia en total con la fragmentación interna.

SO: $752 - 751,875 \text{ pag} = 0,125 \text{ pag}$

$$0,125 \text{ pag} \cdot 16 \text{ KiB/pag} = \mathbf{2 \text{ KiB}}$$

P1: $8 - 7,6875 \text{ pag} = 0,3125 \text{ pag}$

$$0,3125 \text{ pag} \cdot 16 \text{ KiB/pag} = \mathbf{5 \text{ KiB}}$$

P2: $19 - 18,875 \text{ pag} = 0,125 \text{ pag}$

$$0,125 \text{ pag} \cdot 16 \text{ KiB/pag} = \mathbf{2 \text{ KiB}}$$

P3: $26 - 25,6875 \text{ pag} = 0,3125 \text{ pag}$

$$0,3125 \text{ pag} \cdot 16 \text{ KiB/pag} = \mathbf{5 \text{ KiB}}$$

P4: $627 - 626,4375 \text{ pag} = 0,5625 \text{ pag}$

$$0,5625 \text{ pag} \cdot 16 \text{ KiB/pag} = \mathbf{9 \text{ KiB}}$$

4. Cuanta memoria recuperaríamos si hubiésemos utilizado páginas de 8 KiB.

SO:	12030 KiB / 8 KiB/pag	= 1503,75 pag	(1504 pag)
P1:	123 KiB / 8 KiB/pag	= 15,375 pag	(16 pag)
P2:	302 KiB / 8 KiB/pag	= 37,75 pag	(38 pag)
P3:	411 KiB / 8 KiB/pag	= 51,375 pag	(52 pag)
P4:	10023 KiB / 8 KiB/pag	= 1252,875 pag	(1253 pag)

SO: $1504 - 1503,75 \text{ pag} = 0,25 \text{ pag}$
 $0,125 \text{ pag} \cdot 8 \text{ KiB/pag} = 2 \text{ KiB}$

P1: $16 - 15,375 \text{ pag} = 0,625 \text{ pag}$
 $0,625 \text{ pag} \cdot 8 \text{ KiB/pag} = 5 \text{ KiB}$

P2: $38 - 37,75 \text{ pag} = 0,25 \text{ pag}$
 $0,125 \text{ pag} \cdot 8 \text{ KiB/pag} = 2 \text{ KiB}$

P3: $52 - 51,375 \text{ pag} = 0,625 \text{ pag}$
 $0,625 \text{ pag} \cdot 8 \text{ KiB/pag} = 5 \text{ KiB}$

P4: $1253 - 1252,875 \text{ pag} = 0,125 \text{ pag}$
 $0,125 \text{ pag} \cdot 8 \text{ KiB/pag} = 1 \text{ KiB}$

TOTAL 15 KiB
16 KiB/pag 23 KiB



2.6 Gestión de memoria

- **Intercambio** (swapping):
 - Consiste en trasladar temporalmente los procesos **suspendidos** a una unidad de almacenamiento secundario.
 - La unidad de almacenamiento debe ser un dispositivo rápido con espacio para albergar las imágenes de memoria de los procesos (un disco duro es lo idóneo).
 - Es el planificador a medio plazo quien se encarga de decidir que procesos se van a suspender e intercambiar.
 - El **intercambiador** va a ser el módulo del SO encargado de gestionar el espacio de intercambio.

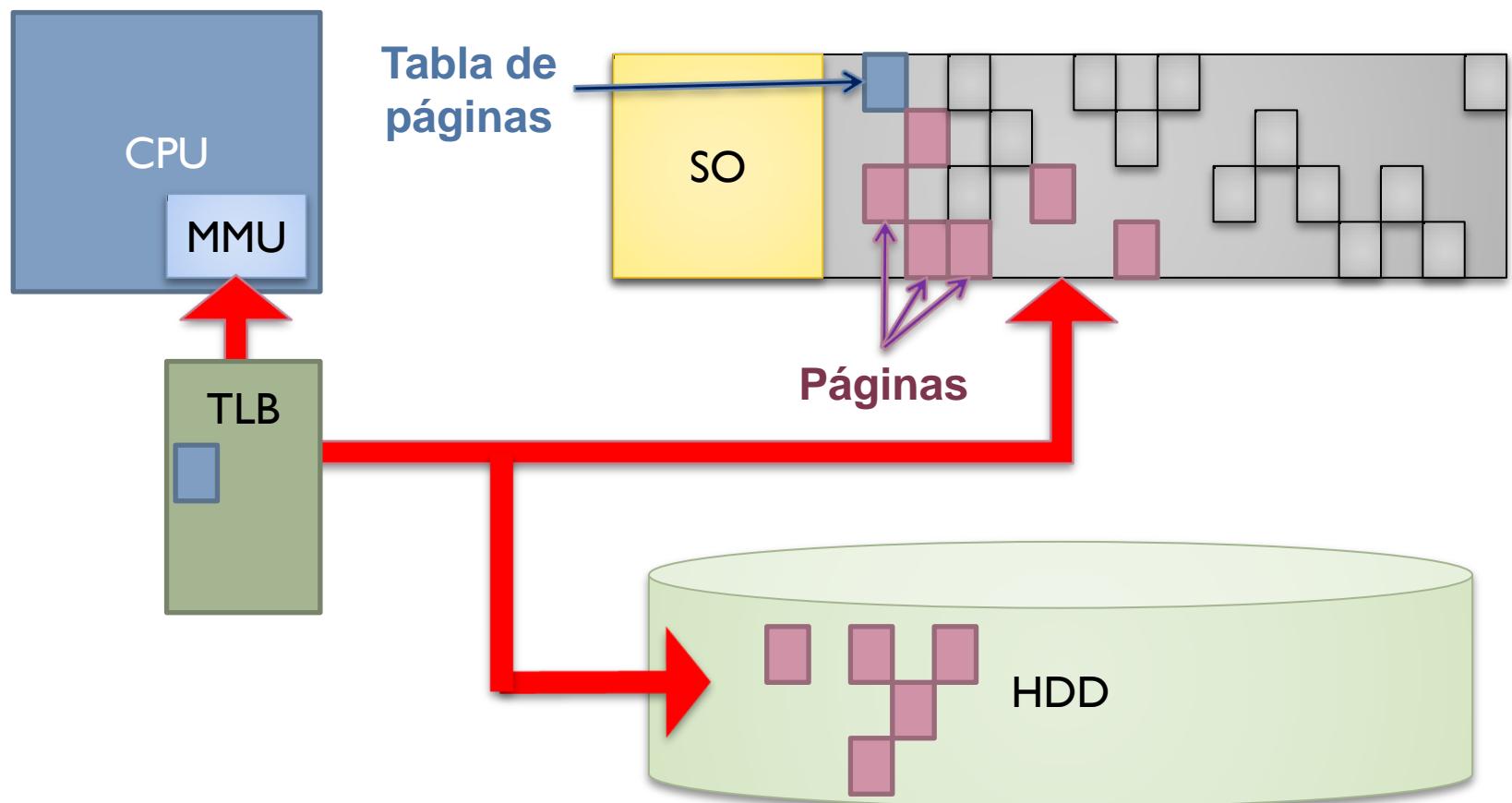


2.6 Gestión de memoria

- Implementación de la memoria virtual en un sistema de memoria paginada:
 - Cada proceso dispondrá de una **tabla de páginas**. Con ella podremos llevar el control de las páginas que están intercambiadas y su localización exacta.
 - Se necesita hardware especial que realice las conversiones de las direcciones de memoria:
 - Unidad de Gestión de Memoria (MMU)
 - Buffer de Traducción Adelantada (TLB)

2.6 Gestión de memoria

- Implementación de la memoria virtual en un sistema de memoria paginada:





2.6 Gestión de memoria

- Implementación de la memoria virtual en un sistema de memoria paginada:
 - **Intercambio Estático** (Linux)
 - Se crea un espacio de memoria fijo para ubicar los procesos intercambiados por el sistema.
 - De su tamaño dependerá el número de procesos intercambiados.
 - Los accesos son rápidos y se suele ahorrar memoria.
 - Linux utiliza una partición (swap) especial para este cometido.
 - **Intercambio Dinámico** (Windows)
 - Se crea un espacio de memoria de tamaño variable para cubrir las necesidades de intercambio de todos los procesos.
 - No impone límites al número de procesos intercambiados.
 - Los accesos son más lentos.
 - Se desperdicia más memoria.
 - Windows utiliza un archivo de paginación (*pagefile.sys*)



2.6 Gestión de memoria

- **Hiperpaginación (*thrashing*)**

- Se origina cuando se lanzan demasiados procesos y el sistema no tiene suficiente memoria RAM para manejarlos.
- El sistema operativo trata de solucionar el problema intercambiando sus páginas a la memoria virtual.
- Se llega un punto en el que las páginas intercambiadas son de uso frecuente.
- El sistema operativo tiene que volver a cargarlas casi inmediatamente.
- El sistema termina gastando casi todo los recursos en el intercambio... y se colapsa.



2.6 Gestión de memoria

- Política **Unused RAM is wasted RAM**
 - En un sistema tradicional multiprogramado sin esta política, (como Windows XP), el sistema intenta agilizarse liberando al máximo la memoria principal, o sea, va a liberar lo más rápido posible los procesos terminados, e intercambiará aquellos que no están previstos utilizarlos durante algún tiempo.
 - En sistemas basados en Linux y en Windows 7, el gestor de memoria va a ocupar la RAM al casi al 100% con procesos que no están en uso.
 - Con ello se asegura la respuesta de procesos que se sospecha que el usuario utilizará próximamente.
- Técnica de **Recolección de Basura**
 - En los sistemas Mac OS X y iOS, un modulo específico del sistema operativo se encarga de liberar aquellos procesos que no se usan demasiado.
 - La liberación de los recursos utilizados por estos procesos es total. Pero se guarda su estado de ejecución en memoria secundaria.
 - Con esta técnica se gana fluidez, pero se pierde rapidez a la hora de solicitar una respuesta a tiempo real.



2.7 Gestión de archivos

- **Unidades de asignación**

- **Sectores físicos (CHS)**

- Un sector es la unidad mínima con la que trabaja el disco duro (no es compatible con otras unidades).
 - Su tamaño depende de la geometría interna del disco duro y no se puede modificar.
 - A cada sector se le asigna tres números: cilindro, cabezal y sector.
 - Su tamaño más usual es de 512 B

- **Sectores lógicos (LBA)**

- Se utilizan para direccionar sectores en unidades de más de 8 GiB, sin importar el tipo.
 - A cada sector se le asigna un número único.
 - Números contiguos podrían no corresponder a sectores físicos contiguos.

- **Bloques o clusters**

- Son la unidad mínima de memoria con la que trabaja el sistema operativo.
 - Su tamaño por defecto es de 4 KiB aunque se puede modificar en el momento del formateo de una unidad de almacenamiento.
 - Al igual que con las páginas de la memoria principal, los bloques lógicos experimentan **fragmentación interna**.
 - Se toman bloques lógicos grandes por razones de eficiencia.



2.7 Gestión de archivos

- **Archivos**

- **Archivo o Fichero (File)**

- Es la colección independiente de datos binarios, agrupados para facilitar su manipulación y almacenaje.
 - Dependiendo del tipo de datos almacenados tendremos diferentes tipos de archivos.
 - El sistema operativo gestiona los archivos mediante un **sistema de archivos**.

- **Sistema de Archivos (Filesystem)**

- Es un conjunto de normas y procedimientos para manipular los archivos en las unidades de almacenamiento.
 - Cada sistema operativo tiene su propio sistema de archivos *nativo*.
 - A veces un sistema operativo puede leer un sistema de archivos de otro sistema operativo.
 - Cada uno de los formatos de almacenamiento óptico tiene su propio sistema de archivos, que suele ser compatible con todos los sistemas operativos.



2.7 Gestión de archivos

- **Archivos**

- **Tipos de archivos:**

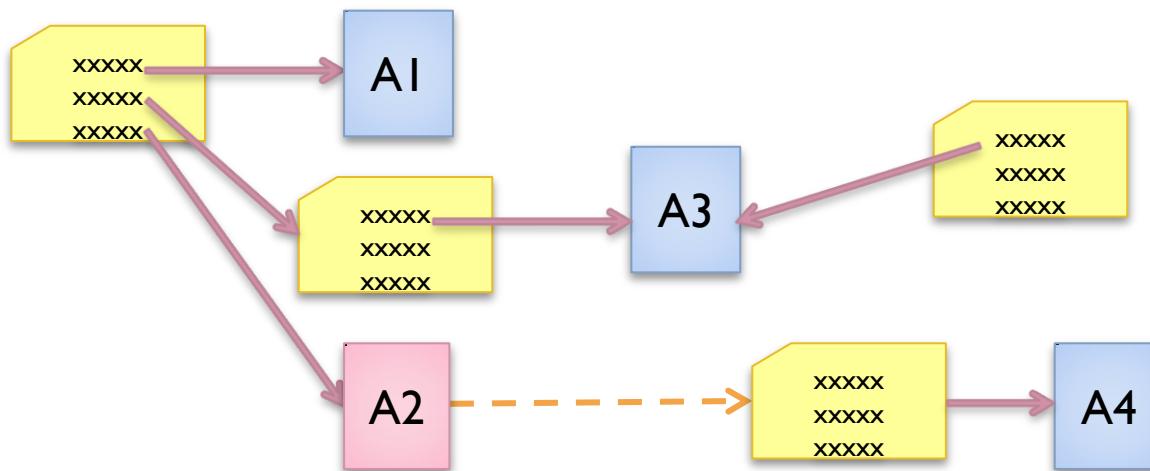
- **Archivos regulares:** contiene información útil para su aprovechamiento por el usuario.
 - **Archivos ejecutables:** contienen programas, o sea, el código de los procesos asociados.
 - **Archivos de datos:** contienen información aprovechable por los usuarios..
 - **Directorios / Carpetas (folders):** contienen información de otros archivos.
 - **Vínculos (links):** contienen la ubicación de otros archivos.
 - **Archivos especiales de E/S (Linux):** Permiten la comunicación con dispositivos:
 - Archivos especiales de cadenas de caracteres → Periféricos E/S
 - Archivos especiales de bloques → Unidades de almacenamiento



2.7 Gestión de archivos

● Estructura de directorios

- **Directorio:** es un archivo con información relativa a la ubicación de otros archivos.
 - Se busca una organización de los archivos para facilitar la manipulación desde el punto de vista del usuario. Éste percibe que los archivos están contenidos dentro de los directorios.
 - En la realidad los archivos están dispersos en la unidad de almacenamiento.
 - La manera a la que se referencia un archivo se le llama enlaces:
 - Enlaces duros, físicos o reales
 - Enlaces blandos, lógicos o simbólicos

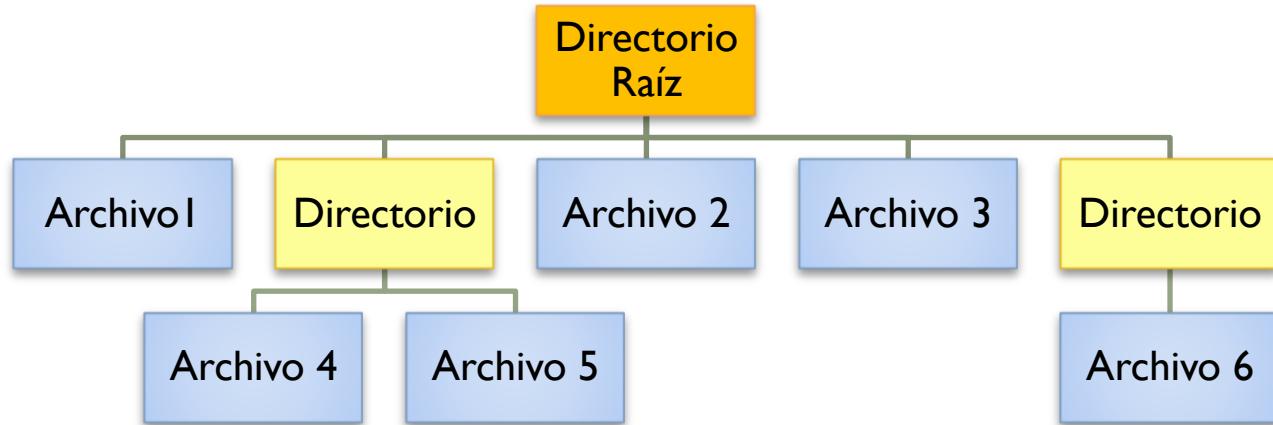




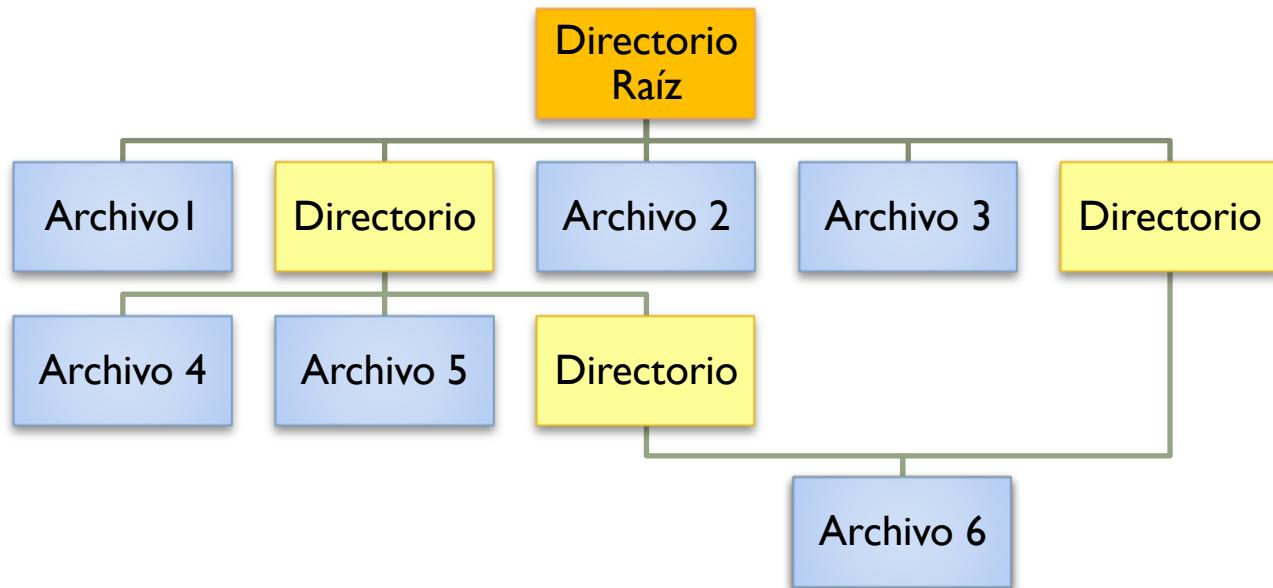
Estructura en un nivel



Estructura en árbol



Estructura en red

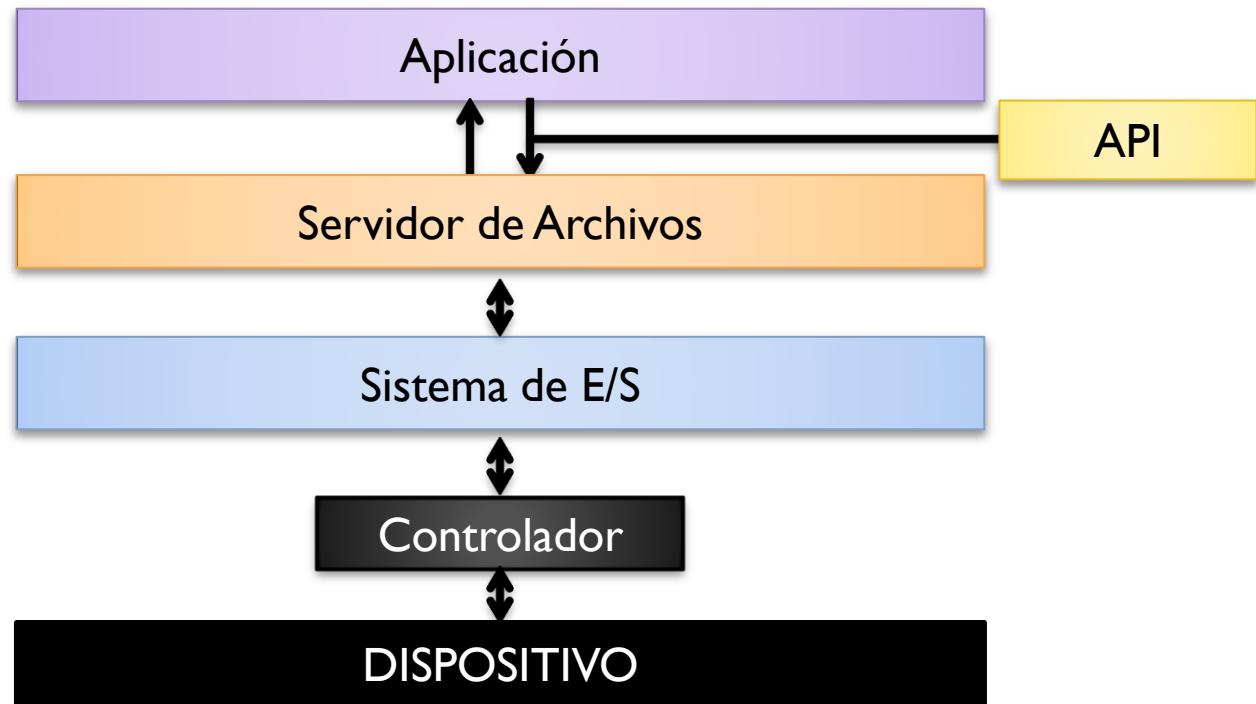




2.7 Gestión de archivos

- **Servidor de Archivos**

- Es la parte del sistema operativo que se encarga de la gestión de todos los archivos accesibles.
- La principal misión es presentar como una estructura ordenada toda la información contenida en las unidades de almacenamiento.



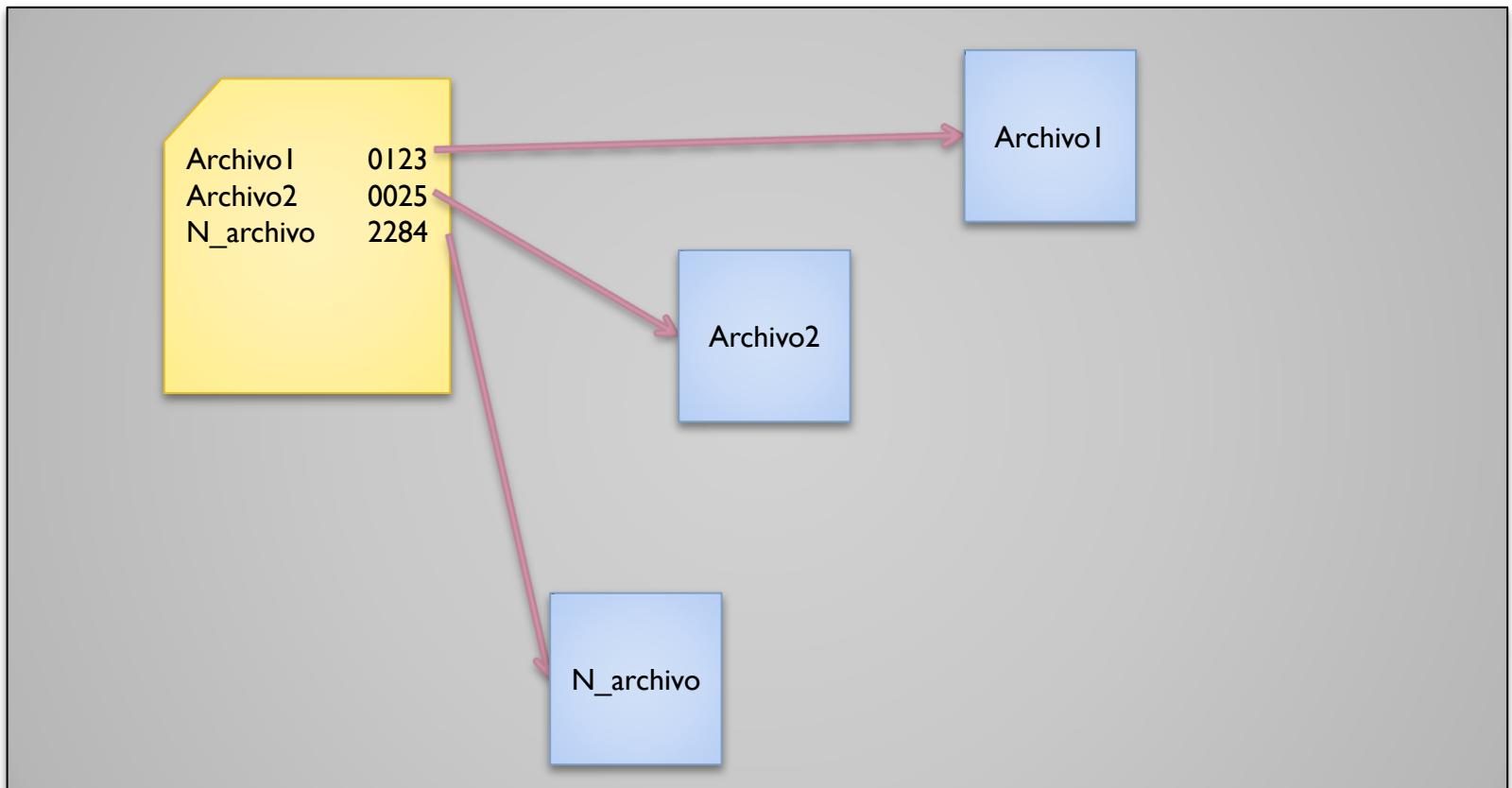


2.7 Gestión de archivos

- **Estructura de directorios**
 - **Operaciones**
 - Gestión
 - **Crear** un archivo o directorio
 - **Borrar** un archivo o directorio
 - **Renombrar** un archivo o directorio
 - **Mover** (cortar-pegar)
 - **Copiar** (copiar-pegar)
 - Obtener/ cambiar **atributos**
 - **Procesamiento**
 - **Abrir** un archivo
 - **Cerrar** un archivo
 - **Leer** un archivo o directorio
 - **Modificar** un archivo o directorio

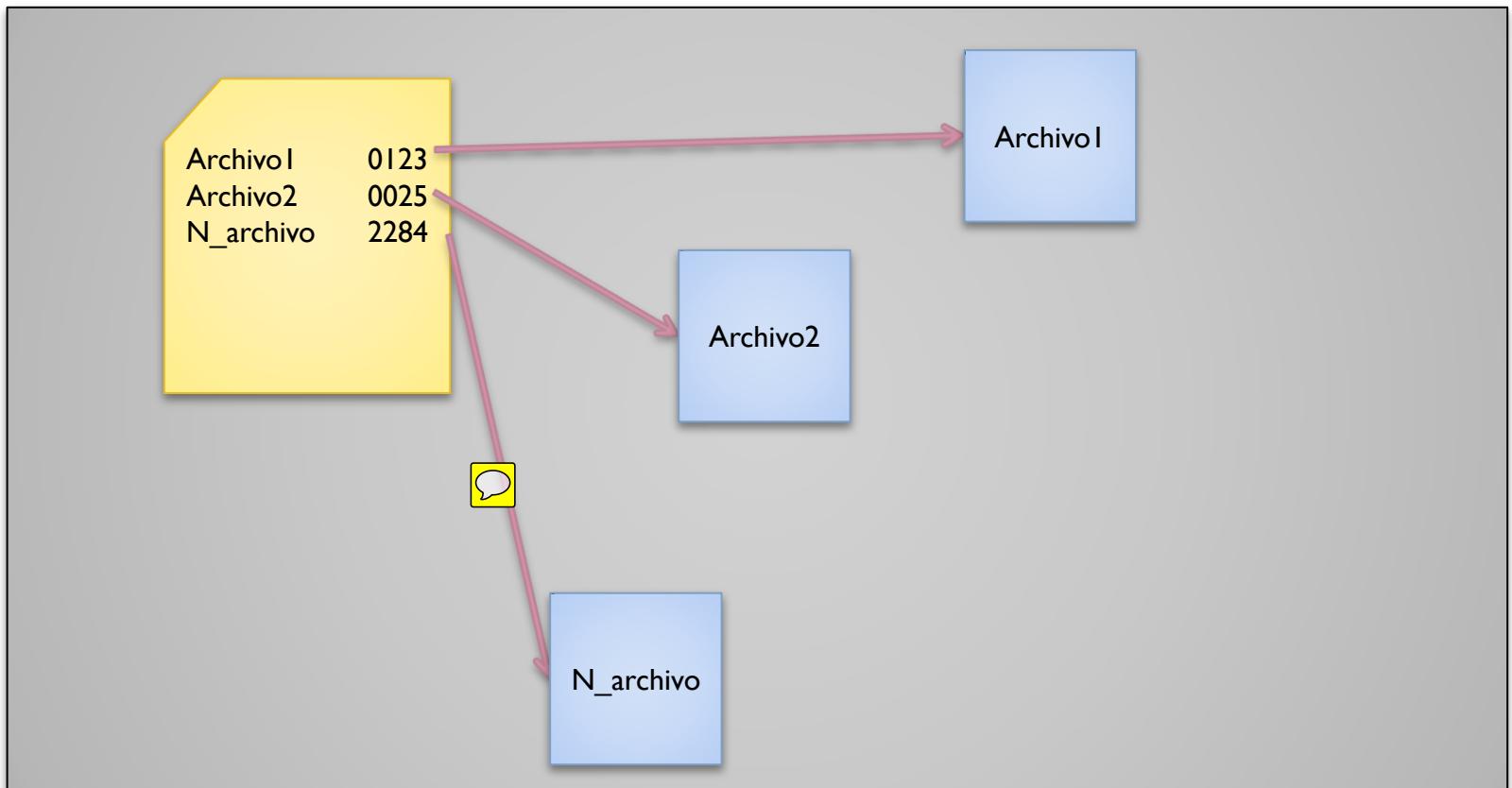
2.7 Gestión de archivos

- **Estructura de directorios**
 - **Operaciones**
 - **Crear** un archivo o directorio



2.7 Gestión de archivos

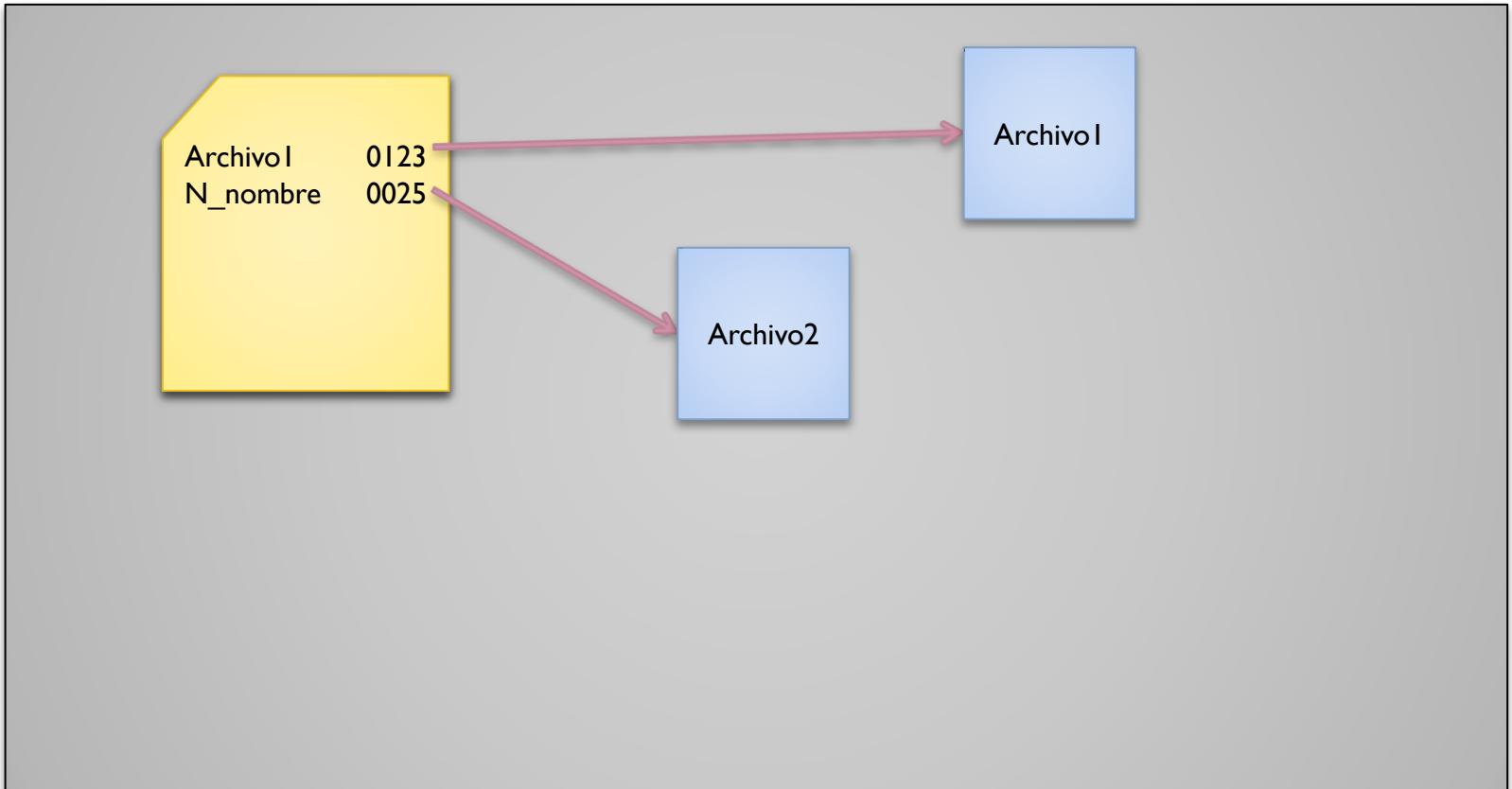
- **Estructura de directorios**
 - **Operaciones**
 - **Borrar** un archivo o directorio





2.7 Gestión de archivos

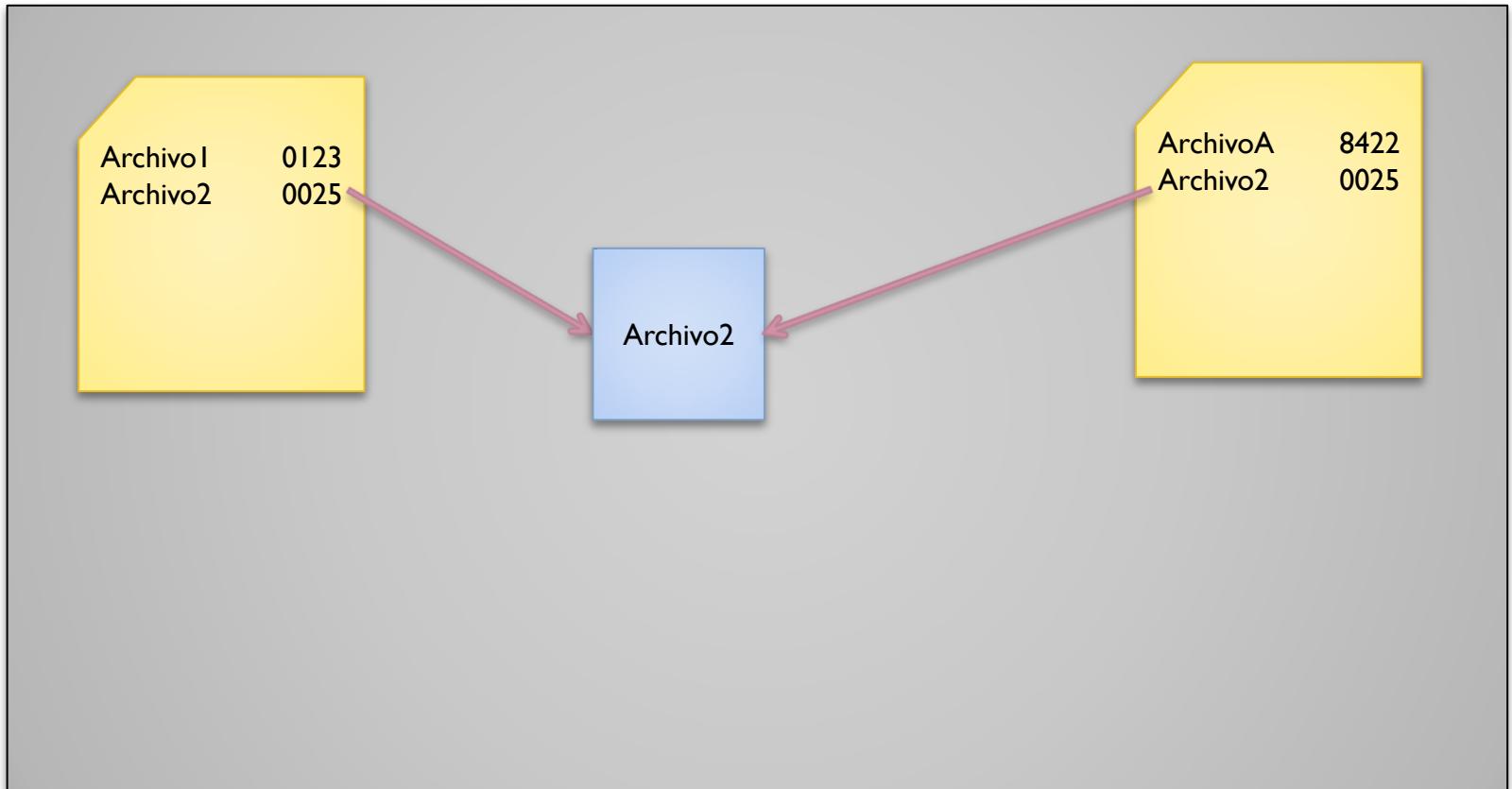
- **Estructura de directorios**
 - **Operaciones**
 - **Renombrar** un archivo o directorio





2.7 Gestión de archivos

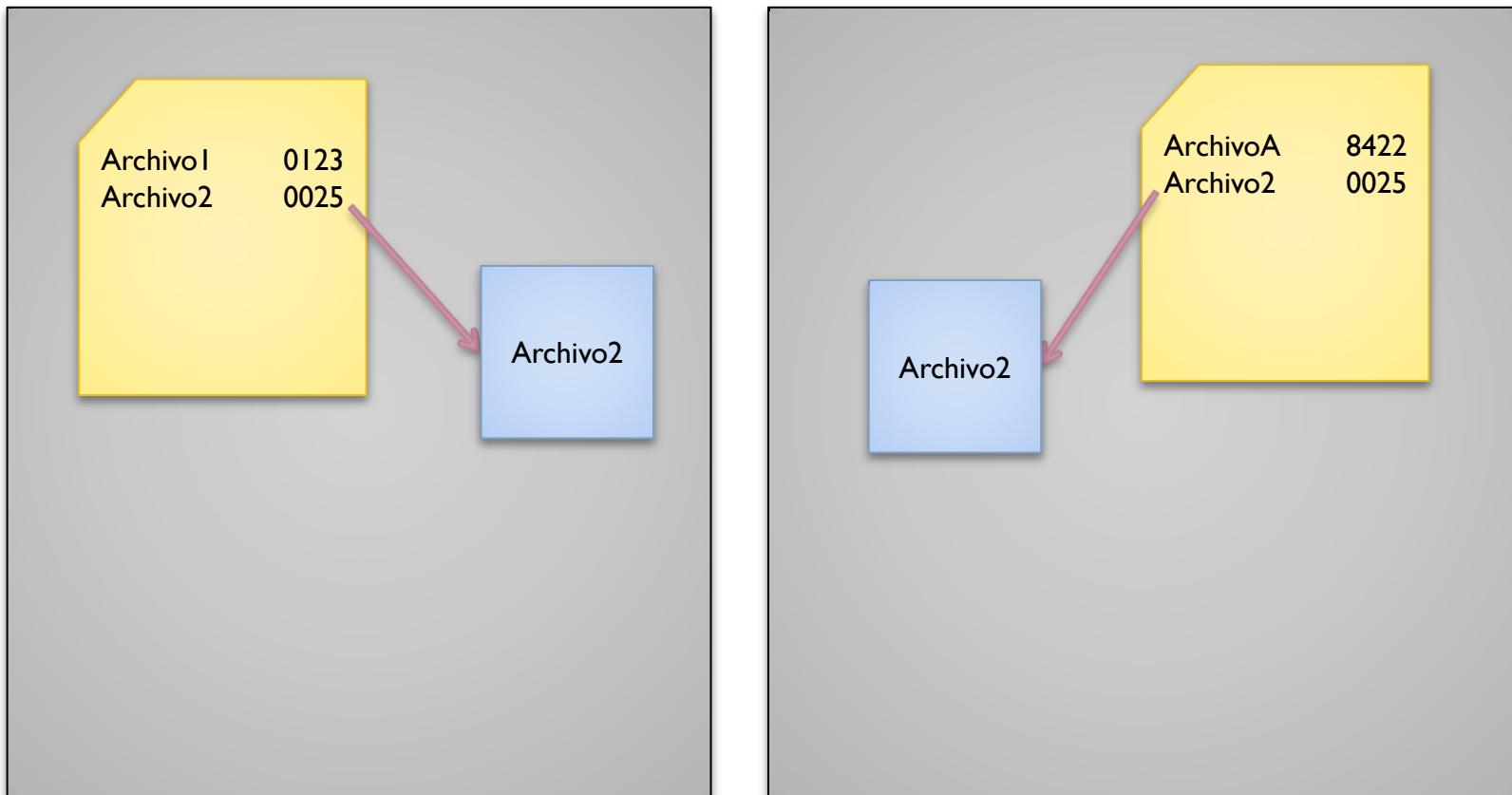
- **Estructura de directorios**
 - **Operaciones**
 - **Mover** un archivo o directorio





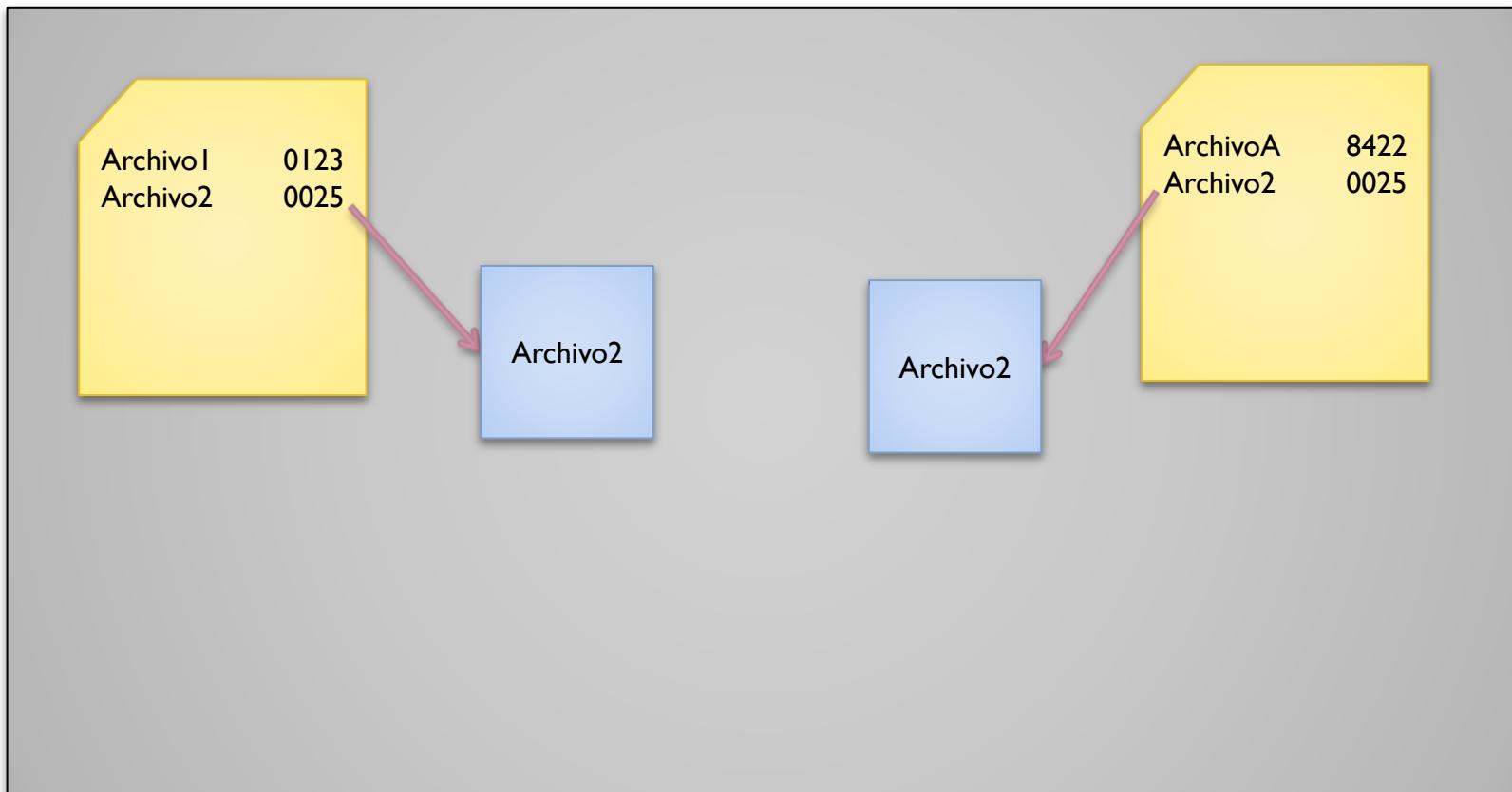
2.7 Gestión de archivos

- **Estructura de directorios**
 - **Operaciones**
 - **Mover** un archivo o directorio



2.7 Gestión de archivos

- **Estructura de directorios**
 - **Operaciones**
 - **Copiar** un archivo o directorio





2.7 Gestión de archivos

- **Estructura de directorio**

- **Operaciones**

- **Propiedades o Atributos**

- **Nombre del archivo**

- FAT16 (Ms-DOS) → 8.3 caracteres en mayúscula (SNF)
 - FAT32 / NTFS (Windows) → 255 caracteres (LNF)
 - HFS+ / ext4 (Linux/Mac) → 255 caracteres

Caracteres no permitidos: " * / :< > ? \ | (en SNF tampoco espacio)

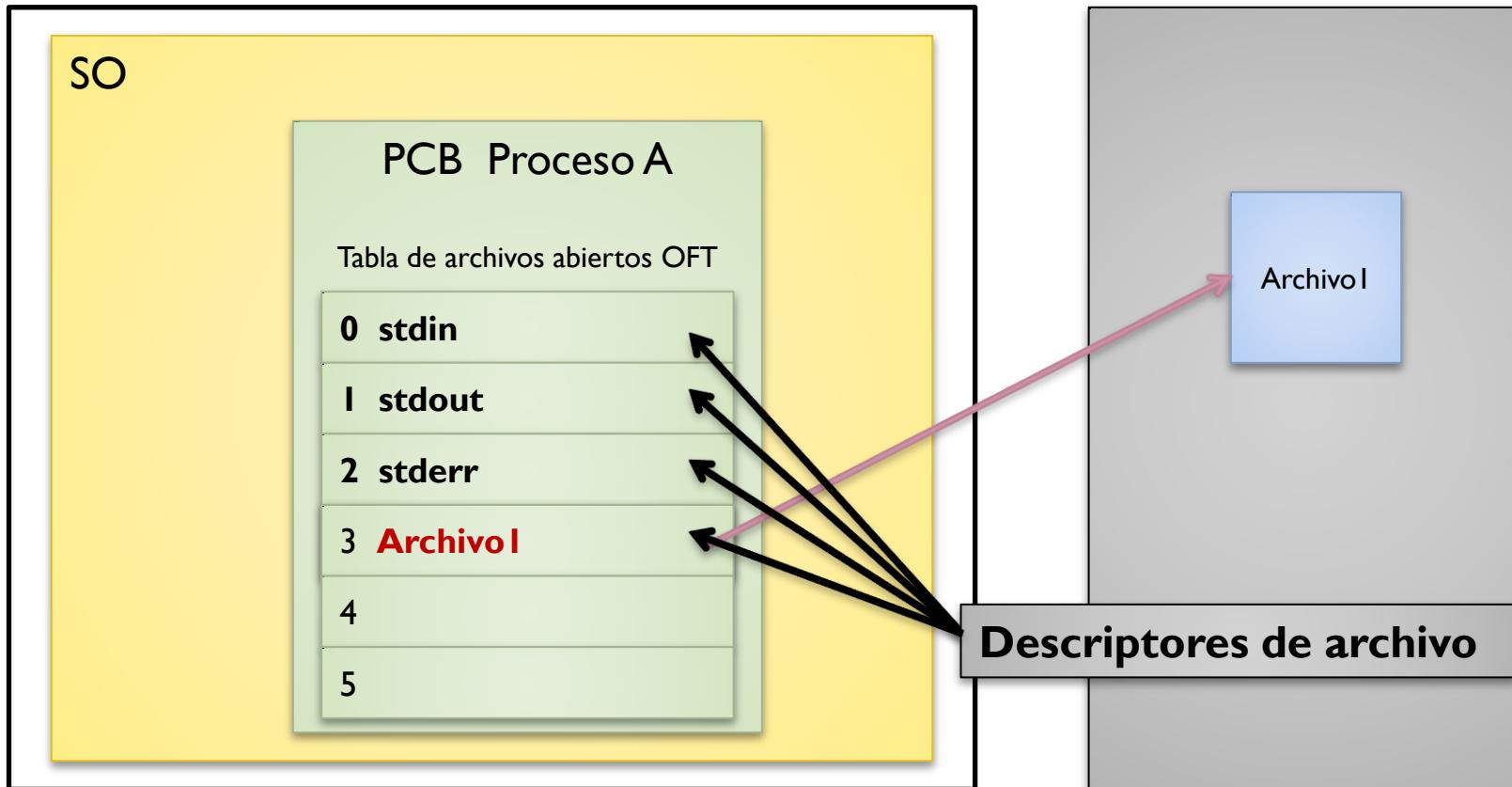
- Tipo de archivo
 - Tamaño y tamaño en disco
 - Permisos
 - Propietario y pertenencia a grupo
 - Momento de creación (Fecha y hora)
 - Momento de la última modificación (Fecha y hora)
 - **Bits (Atributos lógicos)**

- S → Atributo “archivos de sistema”
 - H → Atributo “oculto”
 - R → Atributo “sólo lectura”
 - A → Atributo “modificar”



2.7 Gestión de archivos

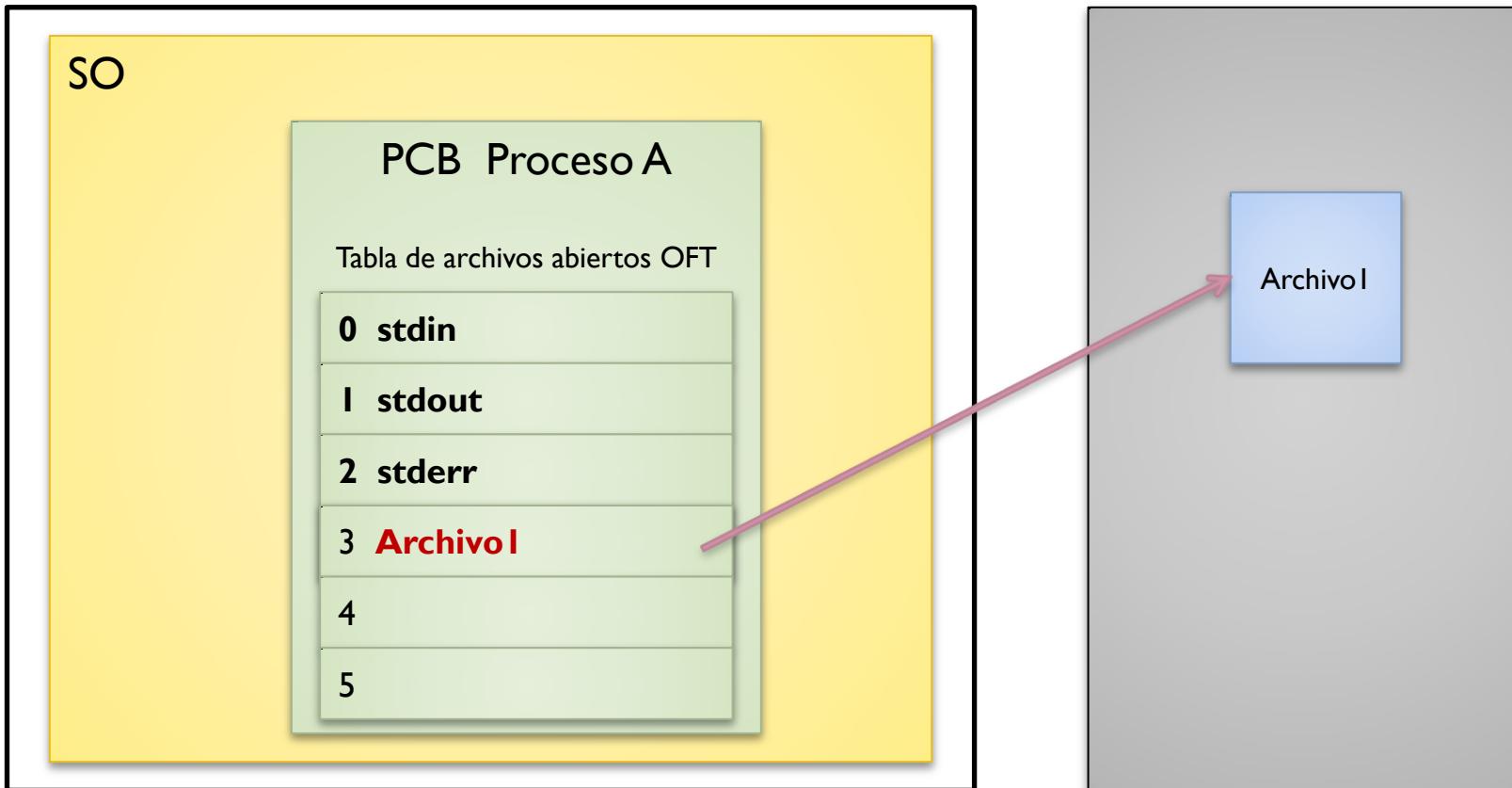
- **Estructura de directorios**
 - **Operaciones**
 - **Abrir** un archivo (llamada **open()**)





2.7 Gestión de archivos

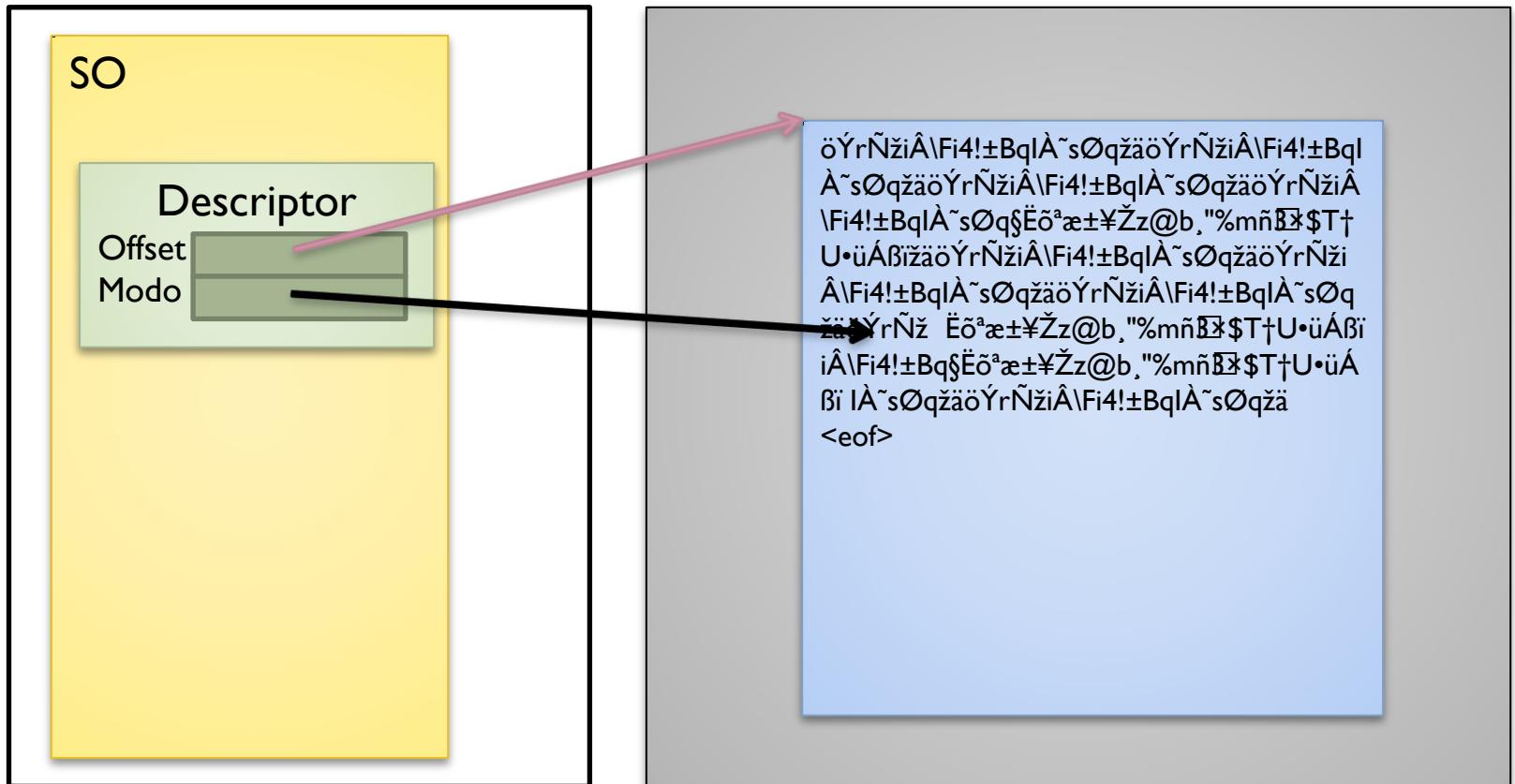
- **Estructura de directorios**
 - **Operaciones**
 - **Cerrar** un archivo (llamada **close()**)





2.7 Gestión de archivos

- **Estructura de directorios**
 - **Operaciones**
 - **Lectura y Escritura** en un archivo (llamadas **read()** y **write ()**)





2.7 Gestión de archivos

- ### Estructura de directorios

- #### Unidades lógicas

- El sistema operativo gestiona los dispositivos de almacenamiento suponiendo una estructura (lógica) que no tiene porque coincidir con la estructura física real.
 - En concreto los discos duros se pueden dividirse en varias **particiones o volúmenes**, siendo cada una de ellas una unidad lógica independiente.
 - Los discos duros también pueden unirse formando unidades distribuidos o formando sistemas **RAID** y comportándose como una única unidad lógica.
 - Otras unidades de almacenamiento no se pueden combinar ni particionar de forma nativa.
 - A veces las unidades lógicas pueden referirse a unidades compartidas de una red externa **NAS o SAN**.



2.7 Gestión de archivos

- **Estructura de directorios:**

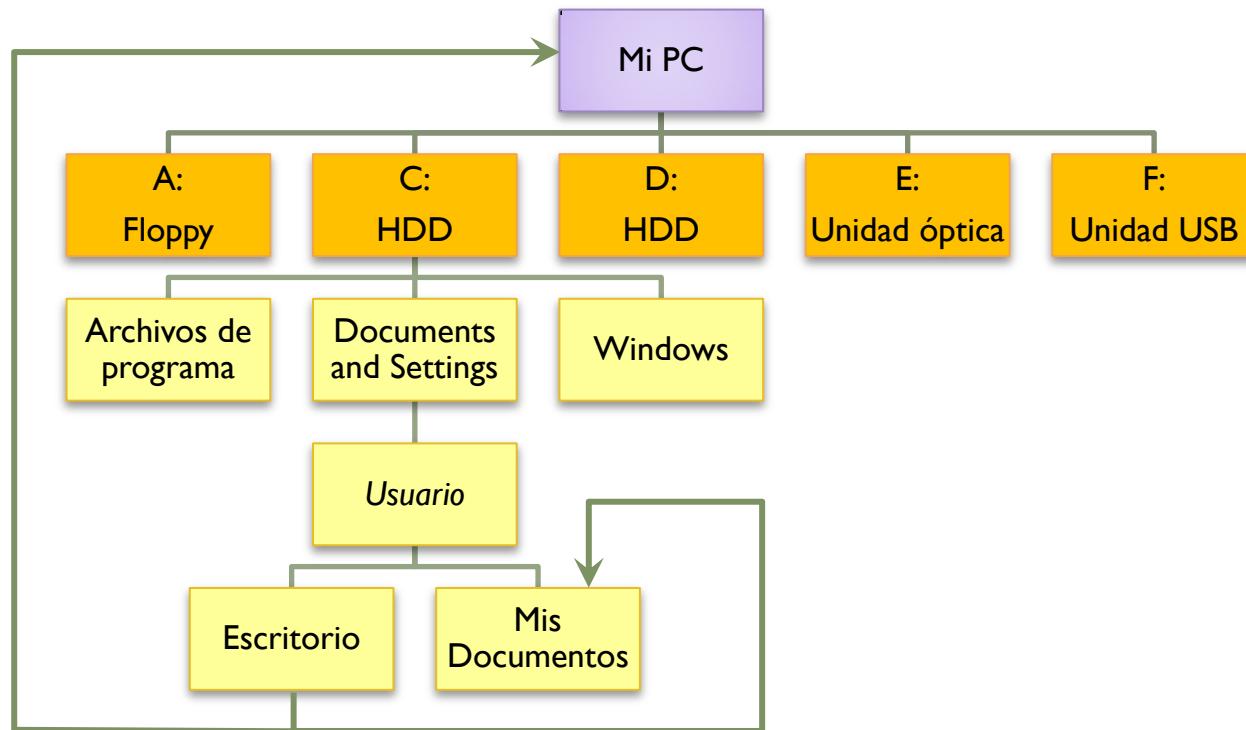
- **Windows**

- Se puede acceder a las unidades lógicas a través de un acceso especial del sistema llamado **Mi PC** o **Equipo**, según la versión.
 - Las unidades lógicas se simbolizan con una letra en mayúscula y dos puntos. Cada unidad tiene una letra diferente. Hay un directorio raíz para cada unidad.
 - **A:** y **B:** se reservan para las unidades floppy.
 - Windows por defecto se instala en la primera unidad de disco duro **C:**
 - Conforme se instalan nuevas unidades, se hace uso de las demás letras en orden alfabético.
 - En la carpeta **Archivos de programa** se copian, por defecto, los archivos de las aplicaciones cuando estas se instalan.
 - La carpeta **Windows** contiene los archivos del sistema operativo.
 - La carpeta **Documents and settings** o **Usuarios** contiene las carpetas personales de los diferentes usuarios.
 - Por otra parte se puede trabajar con una carpeta como si fuera una unidad lógica (comando **SUBST**).



2.7 Gestión de archivos

- Estructura de directorios:
 - Ejemplo de estructura en Windows XP
 - Una unidad floppy
 - Un disco duro con dos particiones,
 - Una unidad DVD-R,
 - Un pendrive conectado vía USB





2.7 Gestión de archivos

- Estructura de directorios:
 - **Sistemas basados en Linux**
 - Toda la estructura de archivos parte de un único directorio raíz simbolizado con el carácter **/**.
 - La carpeta **bin** contiene los ejecutables de los comandos básicos.
 - La carpeta **boot** contiene los archivos necesarios para arrancar el sistema operativo.
 - La carpeta **dev** contiene todos el hardware del sistema representado por archivos de E/S.
 - **fd0** y **fd1** son las unidades floppy
 - **hda**, **hdb**, **hdc**, **hdd** son las unidades físicas IDE del sistema.
 - **hda0**, **hda1**, **hda2**... son unidades lógicas, particionadas a partir de **hda**
 - **sda**, **sdb**, **sdc**... son unidades físicas de SCSI, SATA o USB
 - **sda0**, **sda1**, **sda2**... son unidades lógicas, particionadas a partir de **sda**



2.7 Gestión de archivos

- Estructura de directorios:

- **Sistemas basados en Linux**

- La carpeta **etc** contiene los archivos de configuración del sistema operativo y de las aplicaciones.
 - La carpeta **home** contiene los archivos personales del usuario activo (los otros están ocultos). La carpeta del usuario activo se simboliza como ~
 - La carpeta **lib** contiene las bibliotecas necesarias para que los comandos del sistema funcionen.
 - La carpeta **media** o **mnt** da acceso a la unidades lógicas montadas en el sistema.
 - La carpeta **sbin** contiene los ejecutables demonios.
 - La carpeta **usr** se guardan los archivos de las aplicaciones instaladas.
 - Las carpetas **proc** y **system** son virtuales, las usa el sistema para su ejecución y en realidad no contienen nada.
 - En sistemas basados en Unix se puede “montar” unidades lógicas en una carpeta (comando **mount**).



2.7 Gestión de archivos

- **Estructura de directorios:**

- **Rutas (Path):**

- Es una expresión que sirve para localizar un archivo en una estructura de directorios.
 - **Ruta absoluta:** Ruta desde el directorio raíz.

Windows C:\Windows\System32\write.exe

Linux /home/carlos/Escritorio/carta

- **Ruta relativa:** Ruta desde otro directorio (directorio de trabajo)

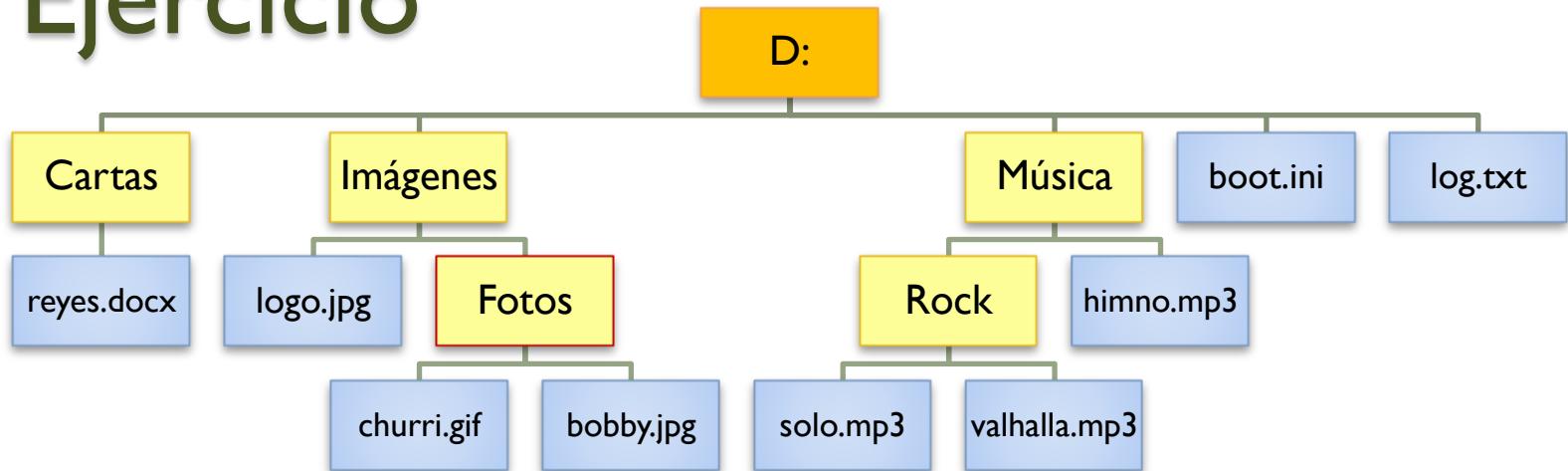
- Carácter . → Representa el directorio actual

- Carácter .. → Representa el directorio padre

Windows ..\Imágenes\foto.jpg

Linux ./almacenaje/datos

Ejercicio



- Indica la ruta absoluta y relativa de los siguientes archivos teniendo en cuenta que el directorio de referencia es “fotos”:

reyes.docx

solo.mp3

log.txt

churri.gif

Imágenes

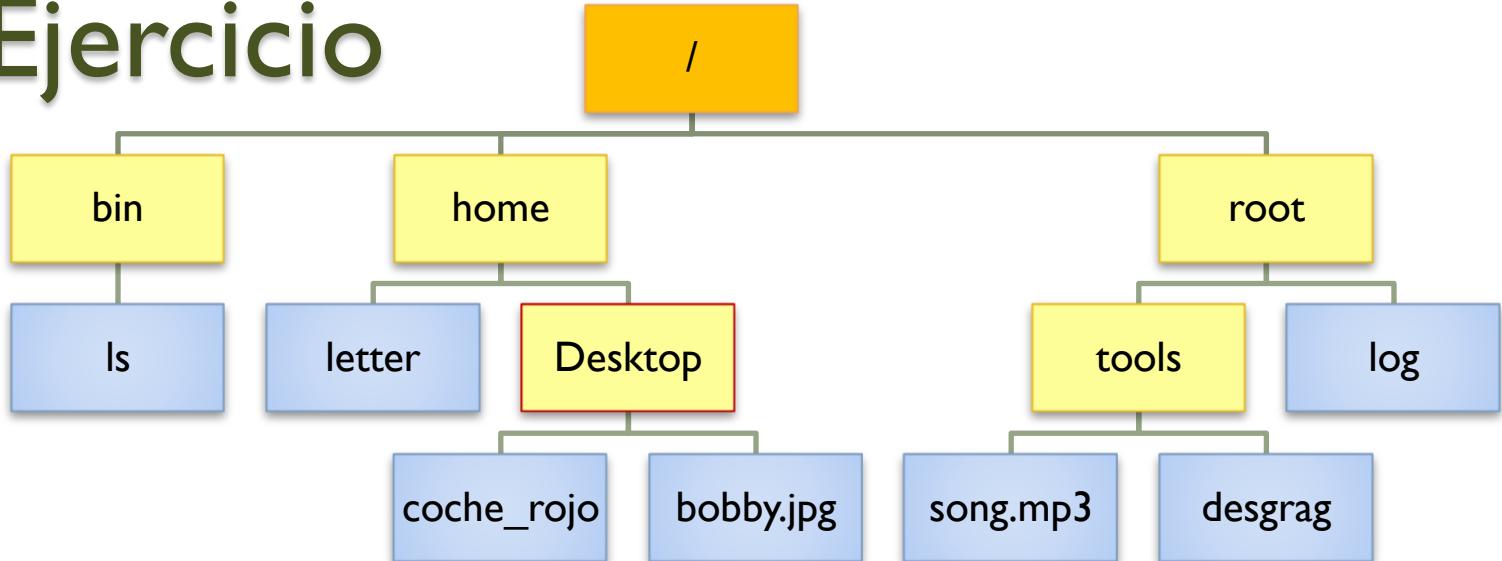
valhalla.mp3

logo.jpg

himno.mp3

Cartas

Ejercicio



- Indica la ruta absoluta y relativa de los siguientes archivos teniendo en cuenta que el directorio de referencia es “Desktop”:

ls

song.mp3

log

coche_rojo

home

desfrag

logo.jpg

letter

tools



2.7 Gestión de archivos

Sistema de Archivos	Tamaño de archivo máximo	Número máximo de archivos	Tamaño máximo de unidad
FAT12 (Microsoft)	32 MiB	4.077	32 MiB
FAT16 (Microsoft)	2 GiB	65.517	2 GiB
FAT32 (Microsoft)	4 GiB	268.435.437	2 TiB
NTFS (Microsoft)	Tamaño unidad	4.294.967.295	16.777.216 TiB
ext2 (GNU/Linux)	2 TiB	Variable	32 TiB
ext3 (GNU/Linux)	2 TiB	Variable	32 TiB
ext4 (GNU/Linux)	16 TiB	Variable	1.073.741.824 TiB
swap (GNU/Linux)	-	-	-
YAFFS (Android)	512 MiB	262.144	8 GiB
HFS (Apple)	2 GiB	65.517	2 TiB
HFS+ (Apple)	8.388.608 TiB	Sin límite	16.777.216 TiB
CDFS ISO9660 (CD)	2 GiB	65.535	Sin límite
UDF (CD/DVD/BD)	16.384 TiB	Sin límite	Sin límite



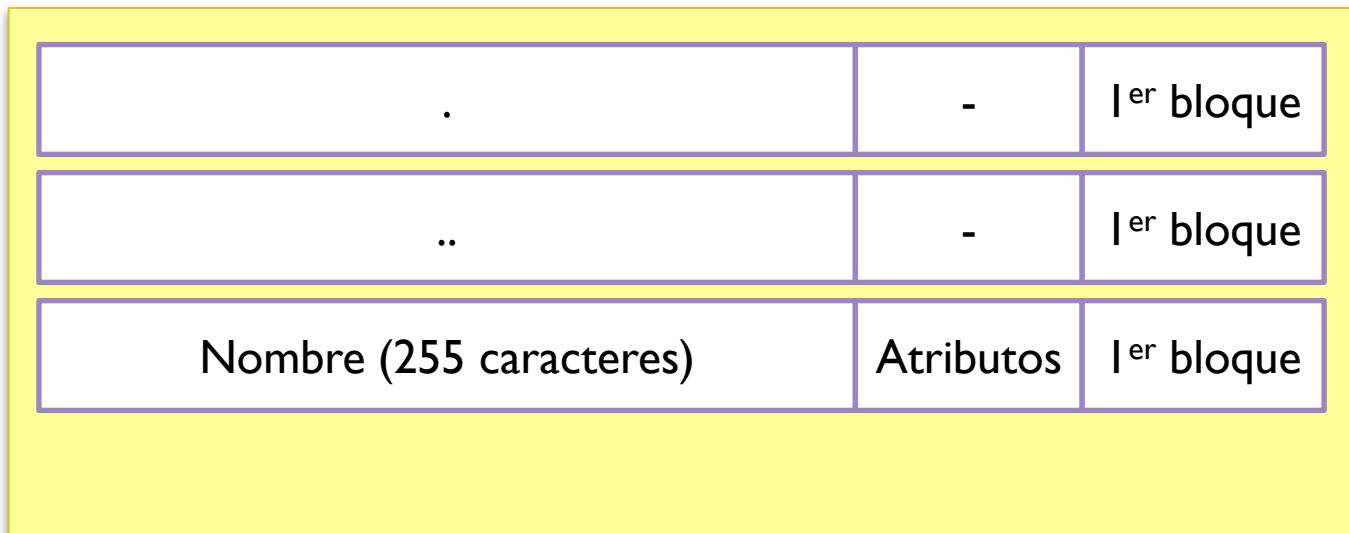
2.7 Gestión de archivos

- **Tabla de localización de archivos FAT32**
 - Es el sistema de archivos nativo de los antiguos Windows.
 - Al formatearse con este sistema, se reserva una sección del disco al comienzo de la unidad para la FAT
 - Sencilla y económica en memoria.
 - La tabla utiliza entradas de 32 bits para gestionar los bloques de los archivos, pero sólo 28 bits se utilizan en realidad.
 - Los archivos se dispersan con facilidad (disminuyendo la eficiencia) con lo que es necesario desfragmentar la unidad con frecuencia.
 - Muy poco resistente a los errores. Por lo que el sistema de archivos esta duplicado como copia de seguridad.
 - No almacena información sobre permisos de usuario.
 - Recomendada para unidades de almacenamiento pequeñas (pendrives).



2.7 Gestión de archivos

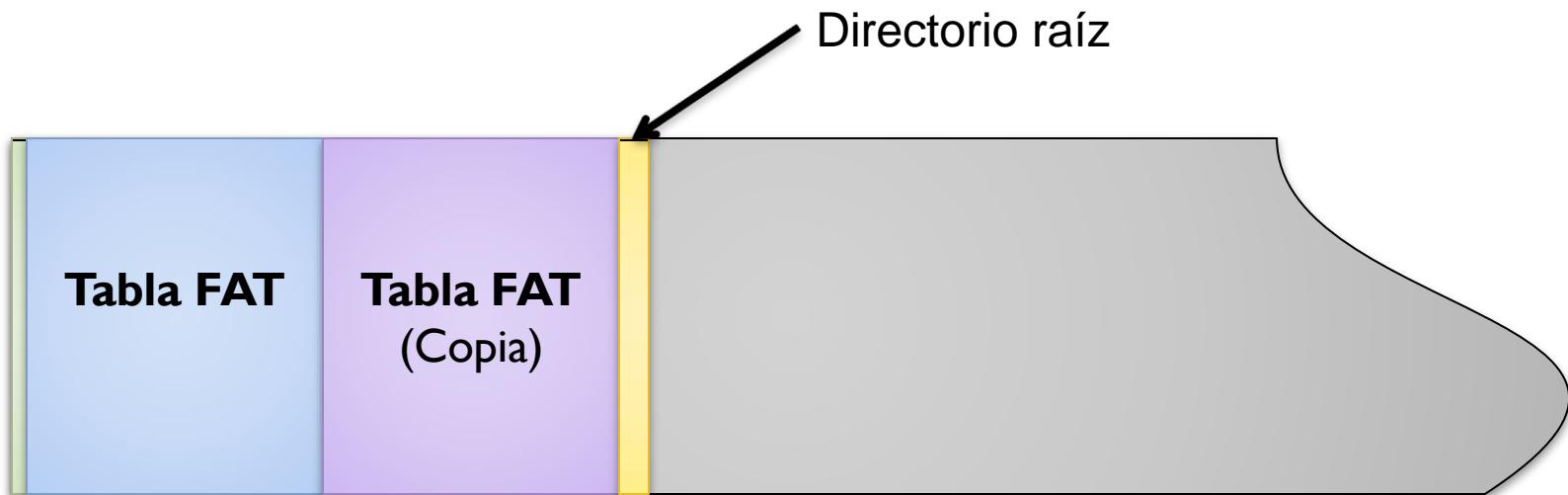
- **Tabla de localización de archivos FAT32**
 - Los nombres de archivo pueden tener hasta 255 caracteres (LNF). En el nombre se incluye la extensión.
 - El tipo de archivo lo determina la extensión. En Windows las extensiones pueden tener más de tres caracteres.
 - Los demás atributos están contenidos en el directorio.





2.7 Gestión de archivos

- **Tabla de localización de archivos FAT32**
 - **Estructura**
 - Región reservada
 - Tabla FAT
 - Copia de la tabla FAT
 - Bloques de datos
 - Directorio raíz de la unidad





2.7 Gestión de archivos



Bloque	FAT
...	...
11	
12	
13	14
14	24
15	
16	
17	
18	
19	
20	
21	13
22	
23	eof
24	23
...	...

2.7 Gestión de archivos



- **Tabla de localización de archivos FAT32**

0x00000000: bloque libre

0x00000001: bloque reservado

0x00000002 – 0x0FFFFFEF: bloque usado

0x0FFFFFF0 – 0x0FFFFFF6: bloques reservados

0x0FFFFFF7: marca de bloque defectuoso

0x0FFFFFF8 – 0x0FFFFFFE: bloques reservados

0x0FFFFFFF: marca **eof**.



4 bit sin uso



2.7 Gestión de archivos

- **Sistema de archivos extendido ext4**

- Es el sistema de archivos nativo de la mayoría de los sistemas basados en Linux actuales.
- Consume grandes extensiones de memoria.
- Utiliza una estructura basada en árboles-H de datos capaz de almacenar todo tipo de atributos (metadatos de los nodos).
- La distribución de los bloques en árbol aumenta la eficiencia en los accesos.
- Los bloques contiguos se manejan en grupos (de hasta 2^{15} bloques) llamados *Extents*. Esto aumenta el rendimiento y disminuye la dispersión de los bloques por el disco duro.
- Es muy resistente a los errores. Se basa en un sistema transaccional donde las distintas operaciones son atómicas.
- Almacena información detallada sobre la seguridad y los permisos de usuario.
- Recomendada para unidades de almacenamiento grandes.



2.7 Gestión de archivos

- **Sistema de archivos extendido ext4**

- Al formatearse una unidad con este sistema, se reserva un número concreto de bloques para la **tabla de i-nodos** dependiendo del tamaño de la partición.
- Cada i-nodo tiene 128 byte, por lo que cada bloque de la tabla tiene 32 i-nodos.
- Inicialmente los i-nodos no están en uso. Cuando se crea un archivo se marca como usado en el **mapa de i-nodos**. Cuando se borra un archivo, el i-nodo queda libre y se puede reutilizar para otro archivo.
- Del i-nodo parte la estructura de todo el archivo.
- Sólo los atributos, básicos (usuario, grupo, tamaño, momento de creación y modificación...) se guardan en el i-nodo, el nombre del archivo que permanece en el directorio.

.	nº i-nodo
..	nº i-nodo
Nombre (255 caracteres)	nº i-nodo

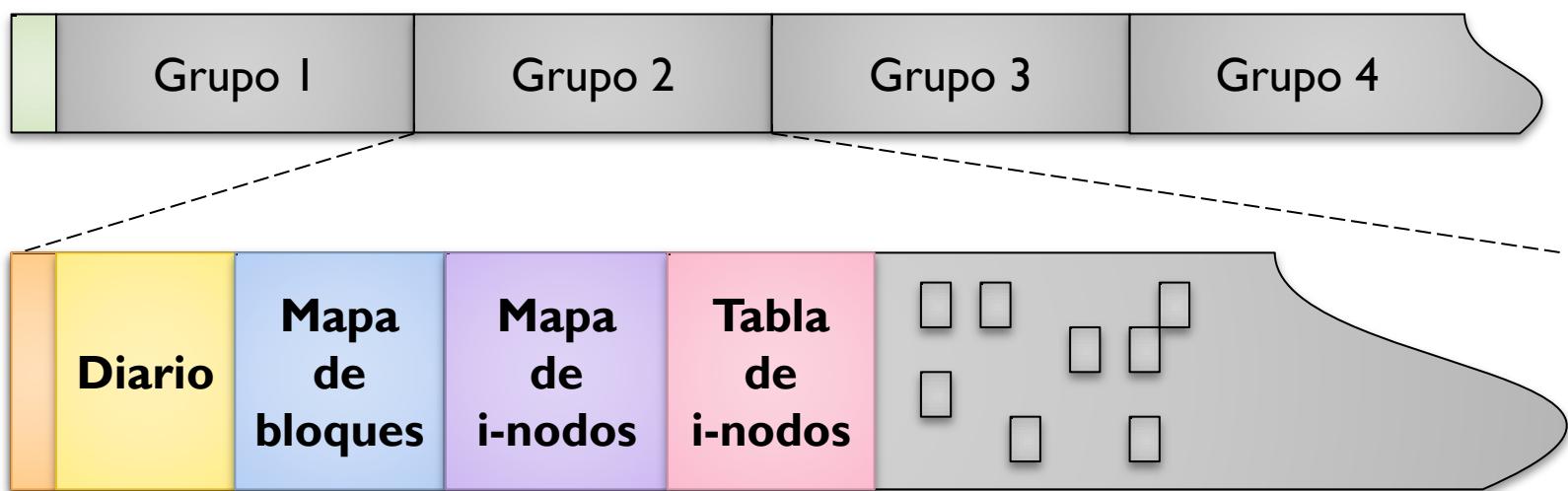


2.7 Gestión de archivos

- **Sistema de archivos extendido ext4**

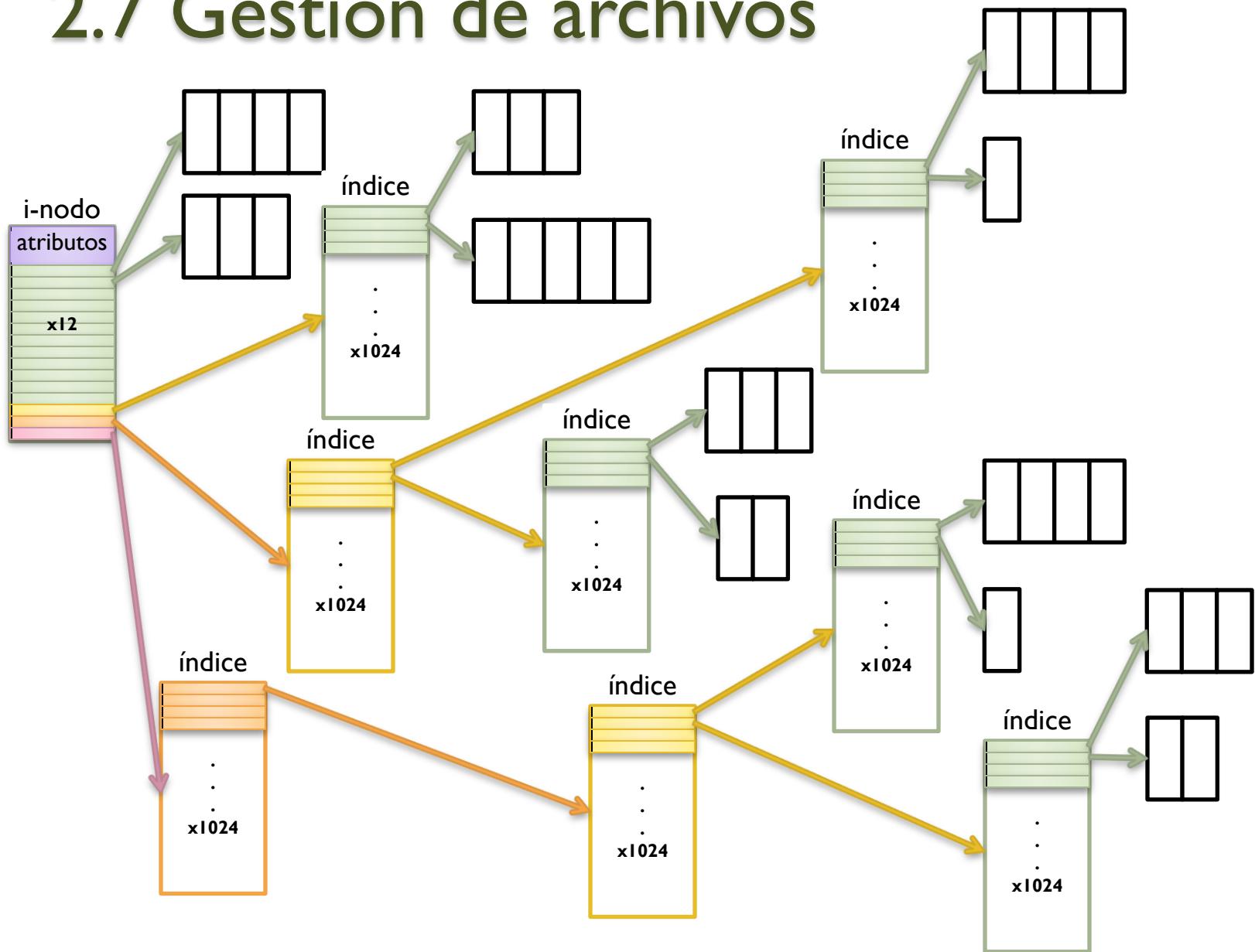
- **Estructura:**

- Las particiones o volúmenes se dividen en **grupos de bloques**.
 - Cada grupo contiene:
 - **Superbloque (descriptor de grupo)**
 - **Diario o Journal**
 - **Mapa de bloques**
 - **Mapa de i-nodos**
 - **Tabla de i-nodos**
 - Bloques de datos





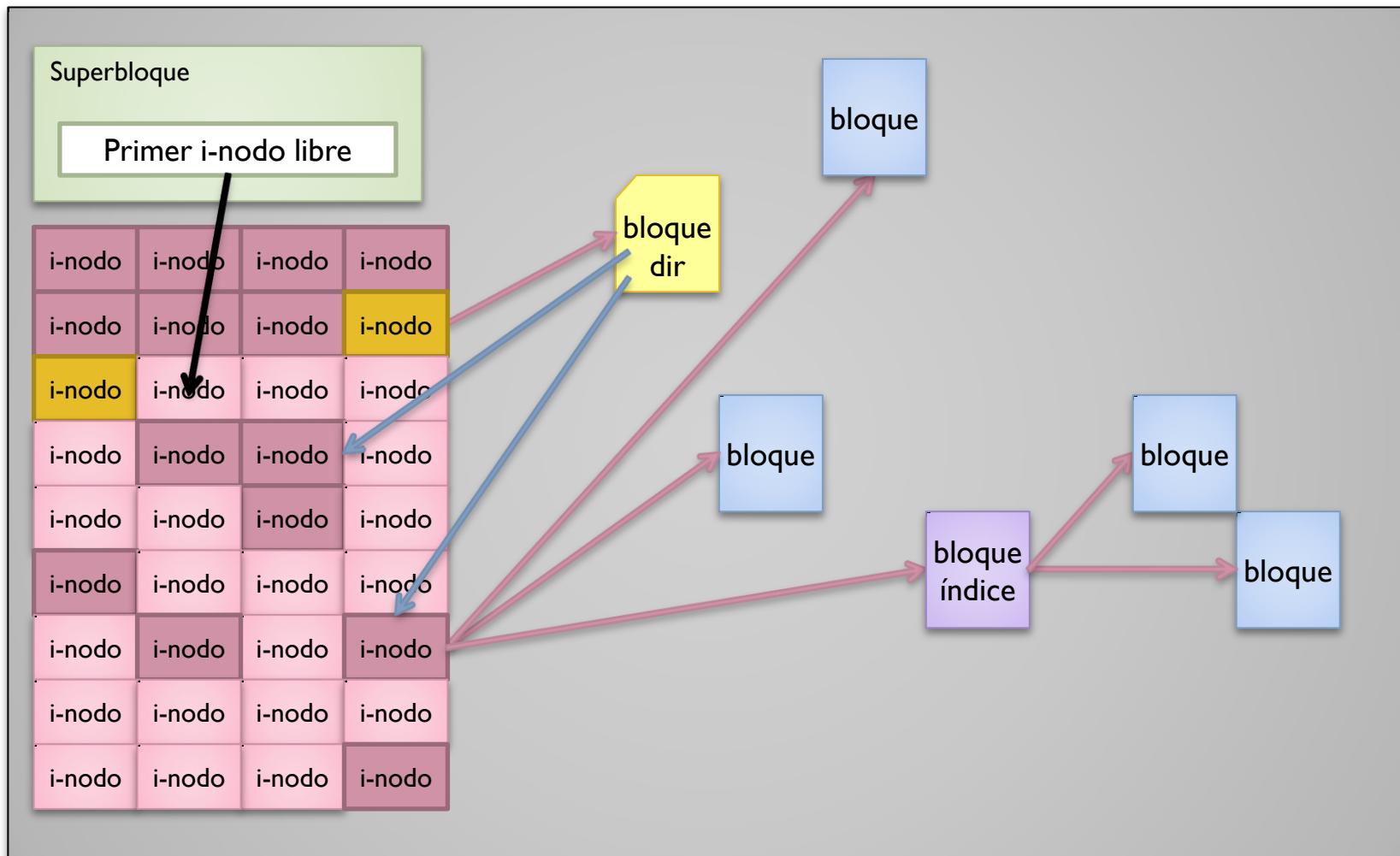
2.7 Gestión de archivos





2.7 Gestión de archivos

- Sistema de archivos extendido **ext4**





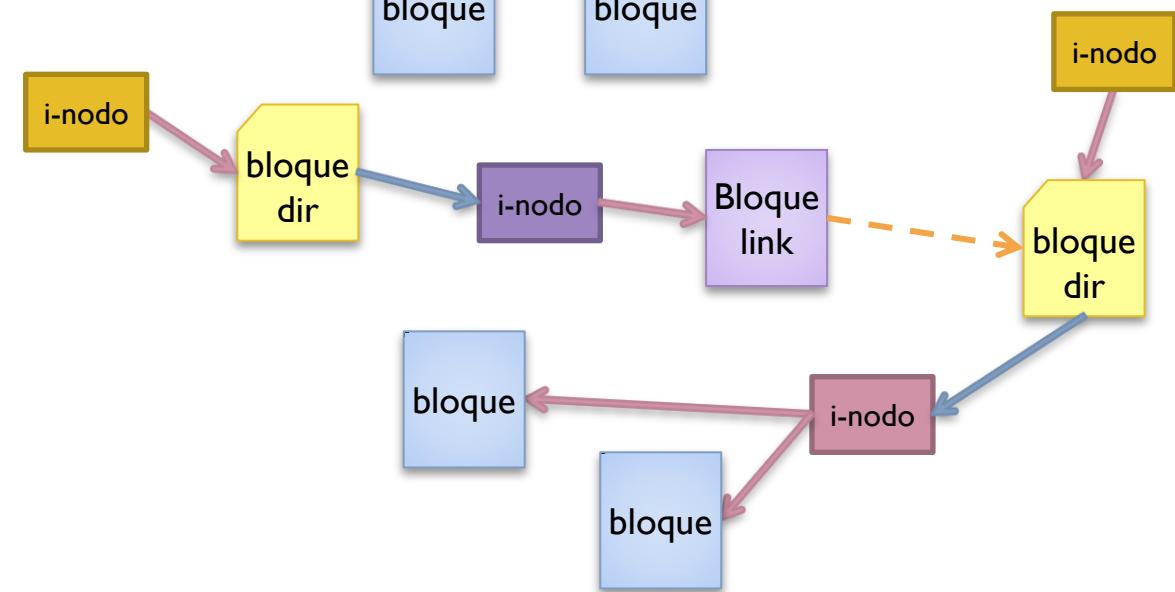
2.7 Gestión de archivos

- **Sistema de archivos extendido ext4**

Enlace duro



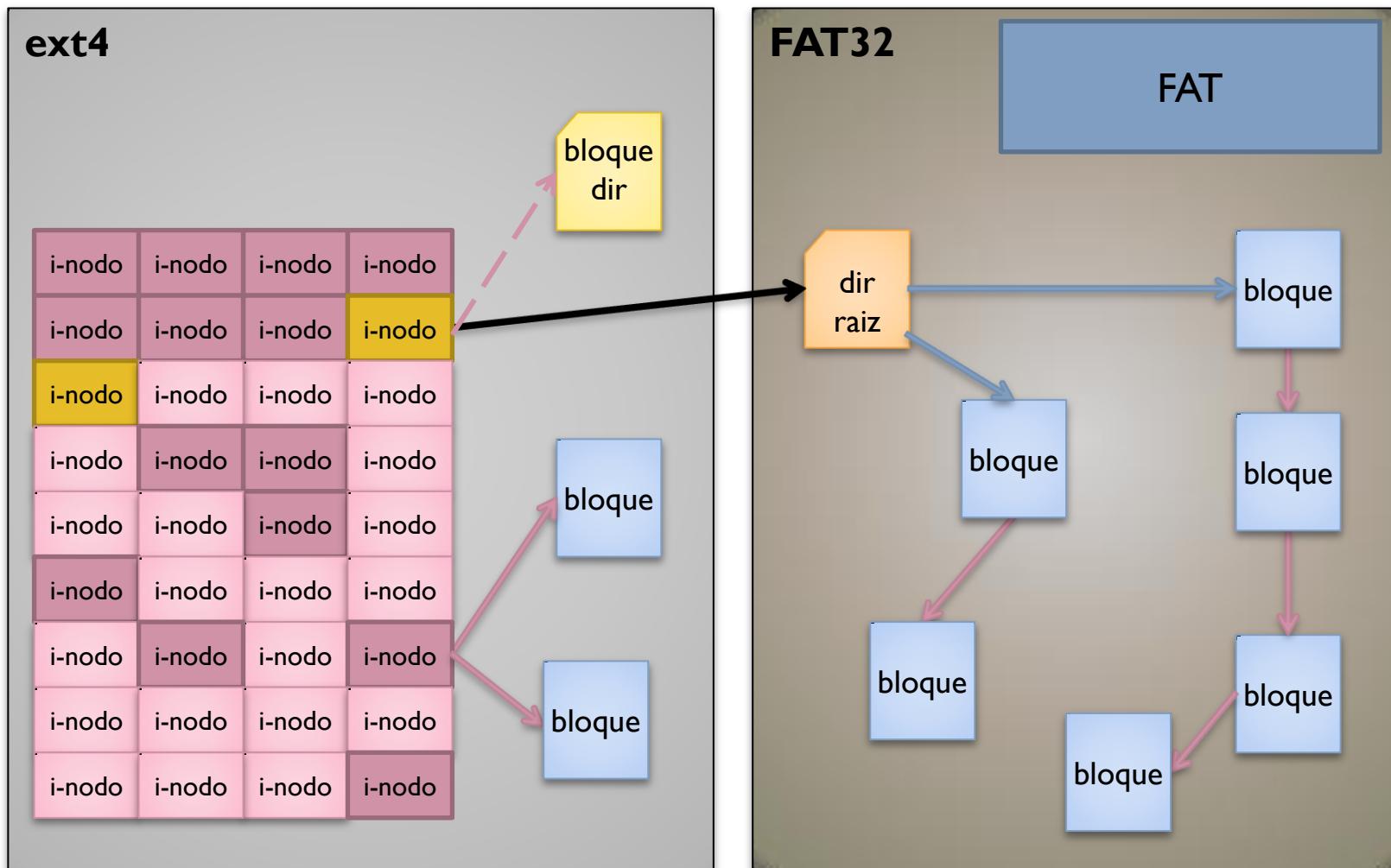
Enlace blando





2.7 Gestión de archivos

- Sistema de archivos extendido **ext4**





2.7 Gestión de archivos

- **Sistema de archivos de nueva tecnología NTFS**
 - Es el sistema de archivos nativo de los modernos Windows.
 - Es mucho más complejo y voluminoso que los sistemas de archivos FAT.
 - Su funcionamiento es un “completo” secreto aunque el proyecto GNU ha podido reproducir algunas operaciones mediante ingeniería inversa.
 - Aunque desde un sistema basado en Linux se puede leer o escribir en un sistema **NTFS**, no se puede hacer ninguna gestión de permisos.
 - Muy resistente a los errores.
 - Permite el cifrado y la compresión de archivos.
 - Recomendada para unidades de almacenamiento grandes.
 - Como tiene prestaciones muy parecidas al **ext4** se sospecha que tiene una estructura y funcionamiento similar.
 - Se sabe que maneja *Extents* de clústeres.
 - Los directorios son tambien muy parecidos a **ext4**



2.7 Gestión de archivos

- **Sistema de archivos de nueva tecnología NTFS**
 - Utiliza archivos (meta-archivos) para gestionar su funcionamiento, contenidos en el directorio **System Volume Information**.
 - Se sabe que posee un **Archivo de registro** con una función similar al **Diario de ext4**.
 - Se sabe que posee un **Archivo de volumen** con una función similar al **Superbloque de ext4**.
 - Se sabe que posee un **Archivo de atributos** para gestionar los metadatos.
 - Se sabe que tiene un archivo con el mapa de bloques libres.
 - Se sabe que tiene un archivo de *clústeres defectuosos*.
 - Tiene una estructura llamada **Tabla Maestra de Archivos (MFT)**, similar a la **tabla de i-nodos**. Esta tabla no es estática sino que es un archivo capaz de crecer.
 - El equivalente de los i-nodos se llama ahora **registro de archivo**.
 - Los archivos pequeños y directorios no precisan clústeres de datos sino que se almacenan en su totalidad en el **MFT** como si fueran metadatos (esto es extremadamente eficiente).



2.7 Gestión de archivos

- **Universal Disk Format UDF**

- Es el sistema de archivos más extendido en los soportes de almacenamiento óptico.
- Este formato permite la lectura, la escritura y la modificación de los archivos contenidos en discos ópticos reescribibles (RW).
- La capacidad de reescritura suele implicar la instalación de un manejador externo (Nero InCD, Roxio DirectCD...)
- Debido a la cantidad de espacio que se invierte en el sistema de archivos, para **CD-R** se prefiere el sistema **ISO 9660**.
- Se suelen utilizar bloques de 512, 1024 o 2048 bytes.
- **Modos:**
 - **Plano** (Acceso aleatorio/Escritura **track-at-once** o **disc-at-once**)



- **Virtual Allocated Table (VAT)**(Acceso aleatorio/Escritura incremental)



- **Bien Administrado (Spared)** (Acceso y escritura aleatoria)





2.8 Gestión de E/S

- Las aplicaciones acceden a los dispositivos de E/S indirectamente, ya que se utiliza al sistema operativo de intermediario.
 - **Por practicidad:** El programador de una aplicación no debe preocuparse de la arquitectura del hardware. (Capa de Abstracción)
 - **Por protección:** Sólo sistema operativo tiene acceso directamente al hardware y así las aplicaciones no tienen que realizar operaciones criticas (modo kernel).
 - **Por eficiencia:** El sistema operativo planificar los accesos a los dispositivos para una utilización óptima.
 - **Por robustez:** El sistema operativo monitoriza los posibles errores, tratándolos y corrigiéndolos si es posible.



2.8 Gestión de E/S

- El sistema operativo gestiona el tráfico de datos mediante el **Sistema de E/S**.
- El sistema de E/S diferencia entre dos tipos de dispositivos según su exclusividad:
 - **Dispositivos de uso exclusivo:** No pueden ser compartidos por dos procesos a la vez.
 - El sistema de E/S asigna este tipo de dispositivos a los procesos que lo soliciten y cuida de que ningún otro lo utilice.
 - Cuando el proceso finaliza, el dispositivo se libera.
 - **Dispositivos de uso compartido:** Pueden compartirse concurrentemente por varios procesos.
 - El SO debe garantizar el acceso equitativo por más de un proceso.



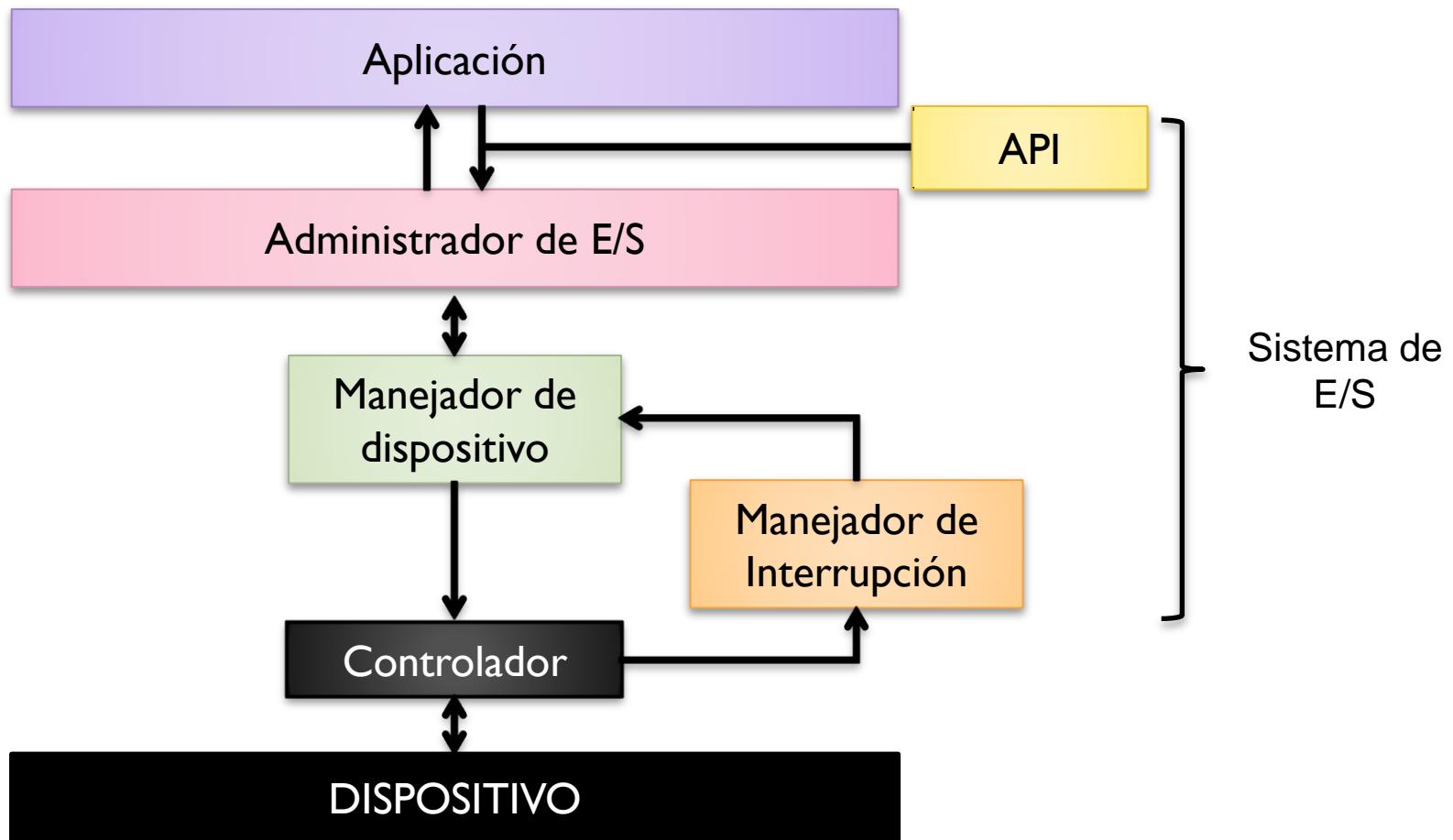
2.8 Gestión de E/S

- ... o según la manera en la que se transmiten los datos:
 - **Dispositivos organizados por bloques (Unidades de almacenamiento):**
 - Manejan información organizada en bloques de un tamaño fijo accesible mediante una dirección.
 - Se trata cada bloque como unidad independiente del resto.
 - Las operaciones sobre los bloques serán de lectura, escritura o búsqueda.
 - **Dispositivos organizados por caracteres (Periféricos):**
 - Reciben y envían los datos como cadenas de bytes o bits sin estructura fija.
 - NO permiten operaciones de búsqueda al no disponer de sistema de direcciones.



2.8 Gestión de E/S

- El **Sistema de E/S** es la parte del sistema operativo que se encarga de esta gestión:





2.8 Gestión de E/S

- **Manejadores de interrupción:**
 - Gestionan las interrupciones hardware generadas por los dispositivos.
 - Activan al manejador de dispositivo adecuado para la gestión
- **Manejadores de dispositivo (drivers):**
 - Son una serie de rutinas que sirven para comunicar cada dispositivo hardware con el sistema operativo.
 - Acceder directamente a los registros de los dispositivos.
 - Cada manejador se comunica con el controlador de un dispositivo.
- **Subsistema independiente del dispositivo:**
 - Presenta una interfaz homogénea para los manejadores de dispositivo.
 - Presenta a las aplicaciones una abstracción de los dispositivos.
 - Protege los dispositivos del acceso no autorizado de los usuarios.
 - Planifica la utilización óptima de los recursos hardware.
- **Interfaces de aplicación (API):**
 - Presentan a las aplicaciones un entorno y un juego de procedimientos para que puedan realizar llamadas al sistema (interrupciones software).



2.8 Gestión de E/S

- **Técnicas hardware para aumentar la eficiencia:**
 - **Acceso Directo a Memoria (DMA):**
 - La CPU normalmente tiene que controlar la transferencia de datos desde un dispositivo a memoria (o viceversa).
 - El DMA es un chip de la placa base que releva a la CPU controlando la transferencia por él.
 - **Presentan un problema:** los DMA utilizan el bus de datos para acceder a la RAM compitiendo con la CPU.
 - **Procesadores de E/S (PE/S) o canales:**
 - Liberan totalmente de operaciones de E/S a la CPU.
 - Establece nuevos buses evitando la competencia con la CPU.
 - **Canal Multiplexor:** maneja la transferencia de varios dispositivos simultáneamente.
 - **Canal Selector:** maneja la transferencia de un dispositivo rápido bloqueando los demás accesos hasta que éste termine.
 - **Caching (Utilización de memorias cache):**
 - Almacén auxiliar de alta velocidad para acelerar el acceso a los datos y evitar cuellos de botella.
 - Se utiliza fundamentalmente para evitar en medida de lo posible el acceso a los dispositivos (por lo general, más lentos).



2.8 Gestión de E/S

- **Técnicas software para aumentar la eficiencia:**
 - **Buffering (Utilización de Buffers):**
 - Almacén auxiliar para permitir que las bajas velocidades de transferencia de algunos dispositivos se compensen con la velocidad de la CPU.
 - Esta técnica no es totalmente efectiva (no se puede satisfacer una demanda masiva).
 - Lo gestiona en el **Administrador de E/S**
 - **Spooling (Utilización de colas)**
 - Cuando varios procesos mandan datos a un mismo dispositivo lento de uso exclusivo, se usa un espacio de memoria, con estructura de cola, donde se almacenan hasta que el dispositivo esta listo.
 - El sistema operativo crea un directorio de spooling y ejecuta un servicio/demonio especial para esta gestión.



2.8 Gestión de E/S

- **Gestión de E/S en Windows XP**

- **El administrador de E/S**

- Es el responsable de todas las operaciones de E/S de Windows.
 - Gestiona los buffers y las cachés.
 - Controla el montaje de los sistemas de archivos.
 - Coordina al Intercambiador (Memoria Virtual)

- **Los manejadores de dispositivo (Drivers)**

- Manejan solicitudes de E/S procedentes del administrador con un formato estándar llamado **Paquete de Solicitud de E/S (IRP)**.

- **El administrador de Plug and Play (PnP)**

- Se coordina con el administrador de E/S para asignar recursos a los dispositivos.
 - Detecta y gestiona a la inserción y eliminación de dispositivos, asignando el driver adecuado cuando se precisa.

- **El administrador de energía**

- Gestiona el encendido y el apagado de los dispositivos.

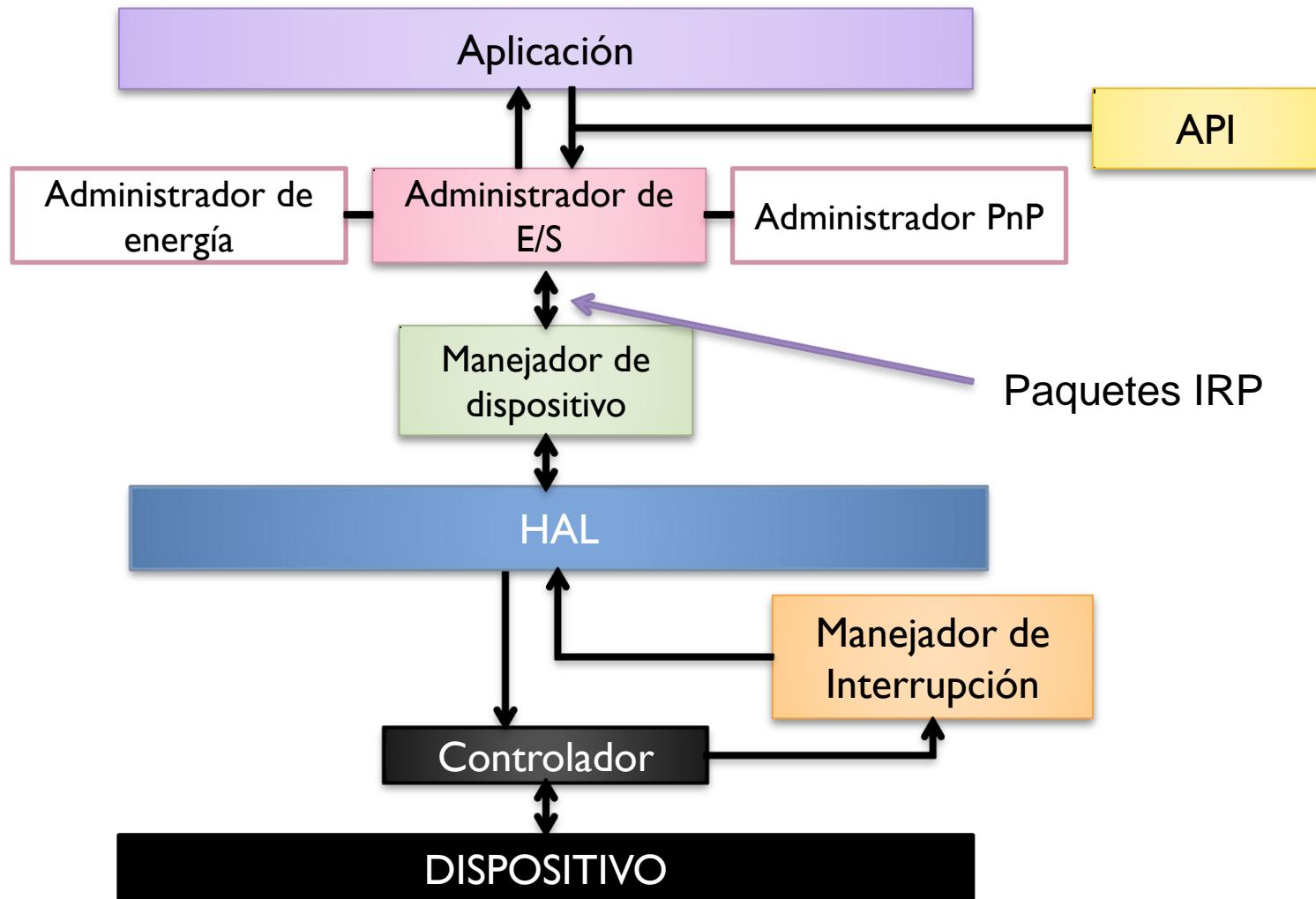
- **La capa de abstracción del hardware (HAL):**

- Capa que aísla a los drivers de los detalles del hardware
 - Es el driver por defecto de todos aquellos dispositivos que no tienen driver propio.



2.8 Gestión de E/S

- **Gestión de E/S en Windows XP**





2.8 Gestión de E/S

- **Drivers en Windows XP**
 - **Windows Driver Model (WDM)**
 - Los drivers en Windows deben ajustarse a unas especificaciones preestablecidas.
 - **El registro**
 - Es una base de datos que almacena información del sistema y las aplicaciones instaladas.
 - Parte de la información que contiene corresponde a los detalles de cada dispositivo y de su correspondiente driver.
 - **Los archivos .inf**
 - Son archivos útiles para la instalación y configuración de los dispositivos.



2.8 Gestión de E/S

- **Gestión de E/S en sistemas basados en Linux**
 - Los dispositivos van a estar representados por archivos especiales de E/S en el directorio /dev.
 - Cada archivo especial de E/S tiene asociado un **driver**.
 - Los **drivers** en Linux están compuestos de dos partes:
 - Una parte se comunicará con el **Administrador de E/S** en modo usuario.
 - Otra parte se comunicará con el Hardware en modo *kernel*.
 - Se accede a los dispositivos con operaciones propias de archivos (*read* y *write*):
 - **Dispositivos de bloques:** Utilizan técnicas de caché para reducir al mínimo el número de transferencias efectuadas.
 - **Dispositivos de caracteres:** Utilizan técnicas de buffer para compensar la diferencia de velocidad (flujos).
 - La gestión del Plug and Play se realiza mediante un demonio (*udevd*).



2.8 Gestión de E/S

- **Gestión de E/S en sistemas basados en Linux**

