Tema 1. Estructuras de SO's

1.1. Arquitecturas monolíticas, en capas, microkernel, y máquinas virtuales.

1.2. Sistemas operativos de propósito específico.

Objetivos

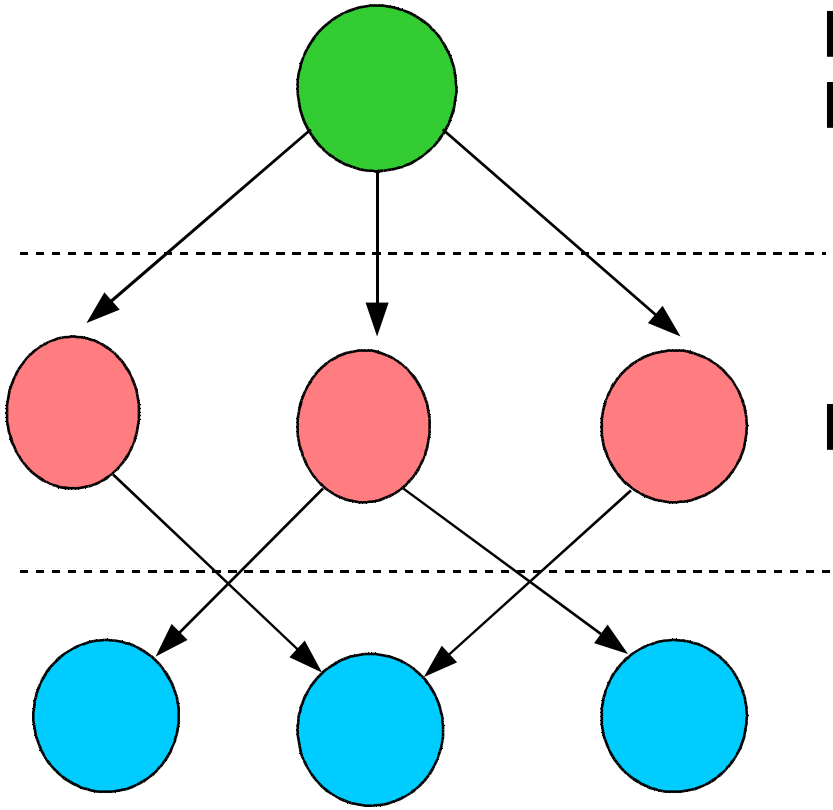
- Conocer diferentes formas de estructurar un SO y los beneficios de cada una.
- Distinguir diferentes tipos de SO´s

1.1. Arquitectura: Sistema Monolítico

- No tiene una estructura bien definida
- El SO está formado por un conjunto de procedimientos de forma que cada uno puede llamar a los demás cuando lo necesite
- Todos se ejecutan en modo supervisor
- SO = un único archivo ejecutable
- No se aplica el principio de ocultación de información

Modelo simple de estructura de un SO monolítico

Todo el código va junto.
Todo está en el mismo espacio de memoria.



Procedimiento Principal

Procedimientos de Servicio

Procedimientos auxiliares

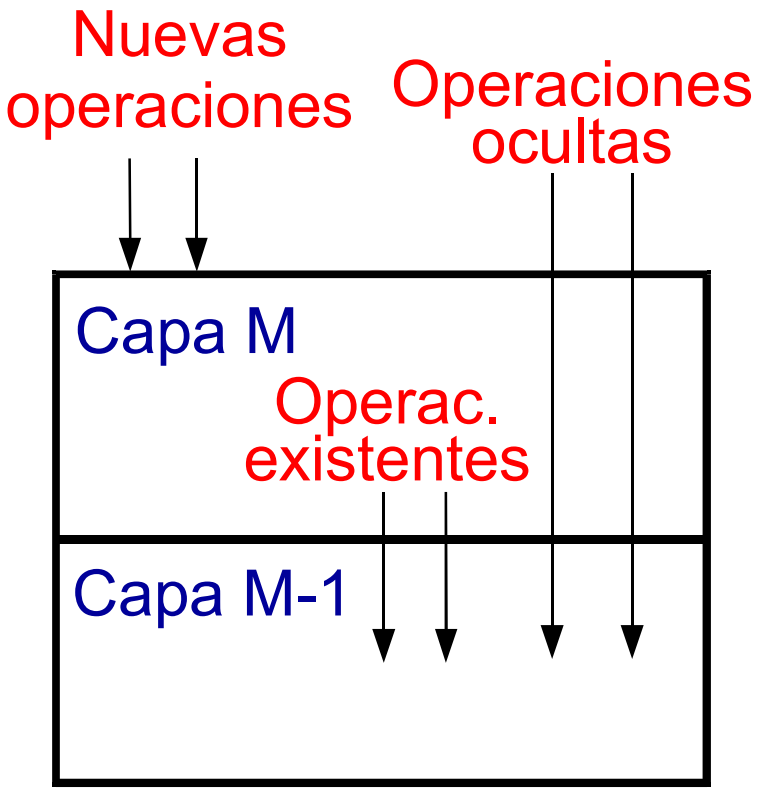
Problemas de los S. Monolíticos

- Son difíciles de comprender y de modificar
- No fiables: un error en alguna parte puede provocar la caída del sistema
- Difíciles de mantener
  - ↘ Desde el principio, los diseñadores han buscado formas de organizar el SO para simplificar su diseño y construcción

Una ventaja es que son rapidísimos porque no hace falta realizar llamadas

Arquitectura: Sistema de Capas

- El sistema se implementa como un conjunto de capas; cada capa es una máquina más abstracta para la capa superior
- Por modularidad, las capas se seleccionan para que cada una utilice funciones sólo de las capas inferiores



Un cambio en una capa no afecta a las capas inferiores

Ejemplo: El Sistema THE

Una capa puede acceder a una inferior sin necesidad de pasar por las intermedias

5: Programas de Usuario
4: Búfering para dispositivos de E/S
3: Manejador de consola del operador
2: Gestión de memoria
1: Planificación de la CPU
Nivel 0: Hardware

- El sistema estaba compuesto de una serie de procesos secuenciales
- Los procesos se sincronizan con declaraciones explícitas de sincronización
- Se puede probar y verificar de forma independiente cada proceso

Su diseño, modificación y depuración es más sencilla.

Problemas de THE

- Los sistemas de capas deben ser jerárquicos pero los sists. reales son más complejos, p.ej.,
  - » El sistema de archivos podría ser un proceso en la capa de memoria virtual
  - » La capa de memoria virtual podría usar archivos como almacén de apoyo de E/S

Es una idealización. Podría darse que una capa necesite de alguna superior

- Sobrecarga de comunicaciones entre procesos de distintas capas

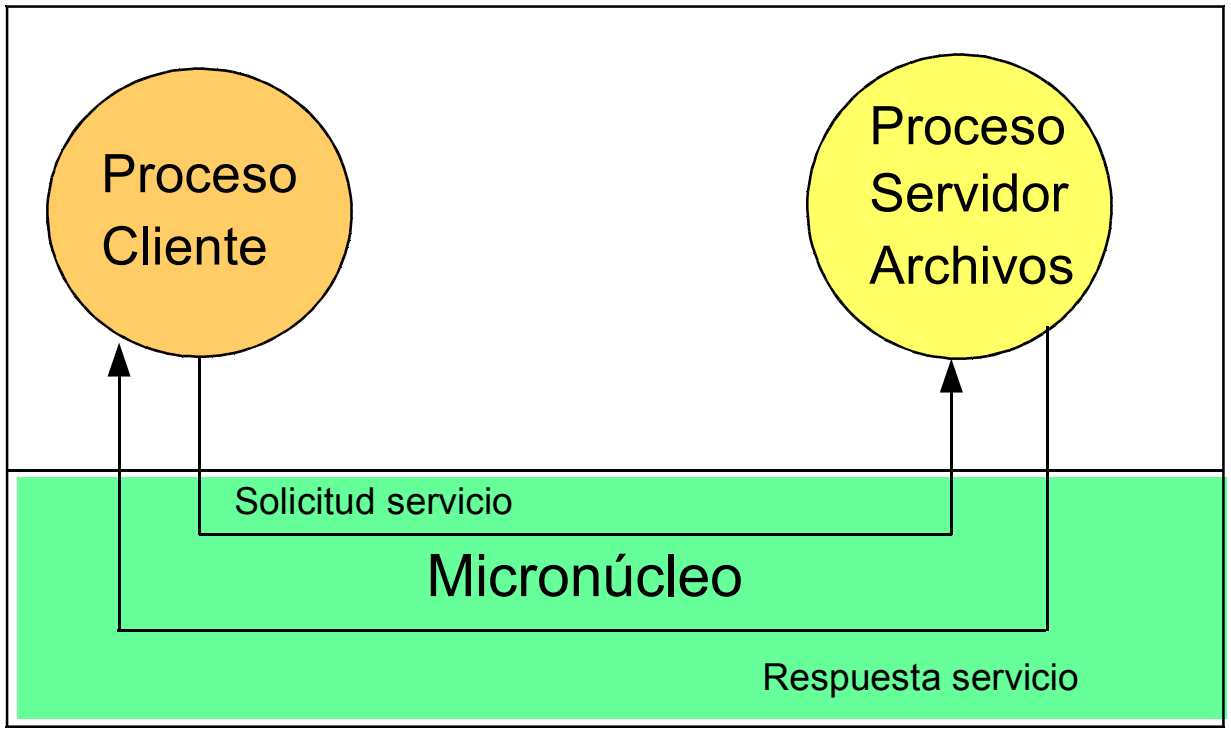
⇒ Esto da lugar a un S.O. más lento

- A menudo, los sistemas están modelados con esta estructura pero no están así construidos

Arquitectura: Microkernel o Micronúcleo

- Reducir el SO a un núcleo mínimo: implementar la mayoría de las funciones del SO como procesos de usuario
  → Mayor flexibilidad
- Para solicitar un servicio, el proceso de usuario (cliente) envía un mensaje al proceso servidor, que realiza el servicio y devuelve al cliente una respuesta
  → Mayor sobrecarga por en envío/recepción de mensajes
- Algunos S. Micronúcleo permiten servidores en modo sistema
  → Más eficiente pero rompe la filosofía micronúcleo
  → Servidores son programas independientes pero se ejecutan en mismo espacio de direcciones del micronúcleo no usan IPCs para comunicarse

Modelo cliente-Servidor

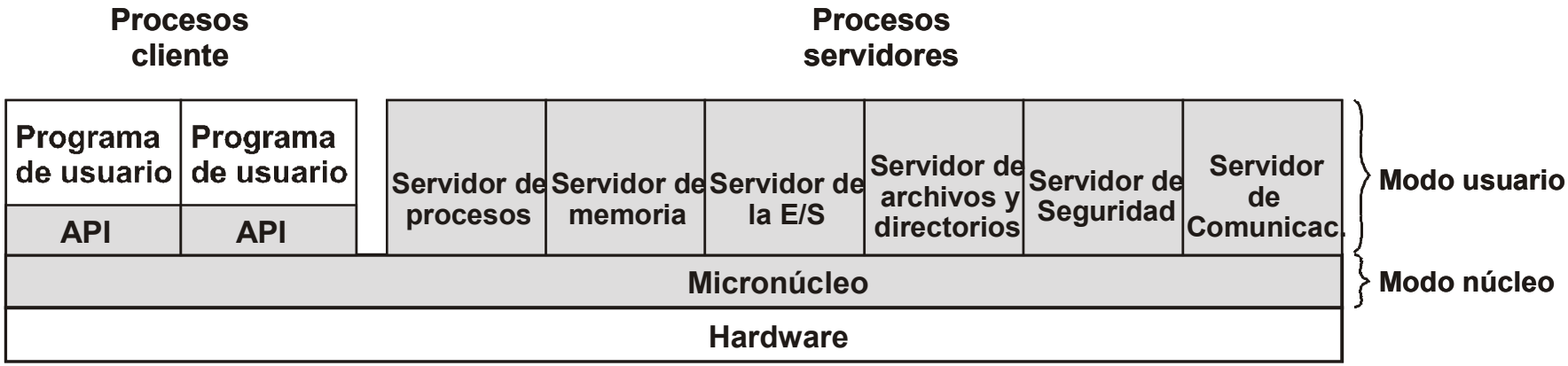


Modo usuario

Modo supervisor

Las llamadas al micronúcleo hacen que no sea tan eficiente.

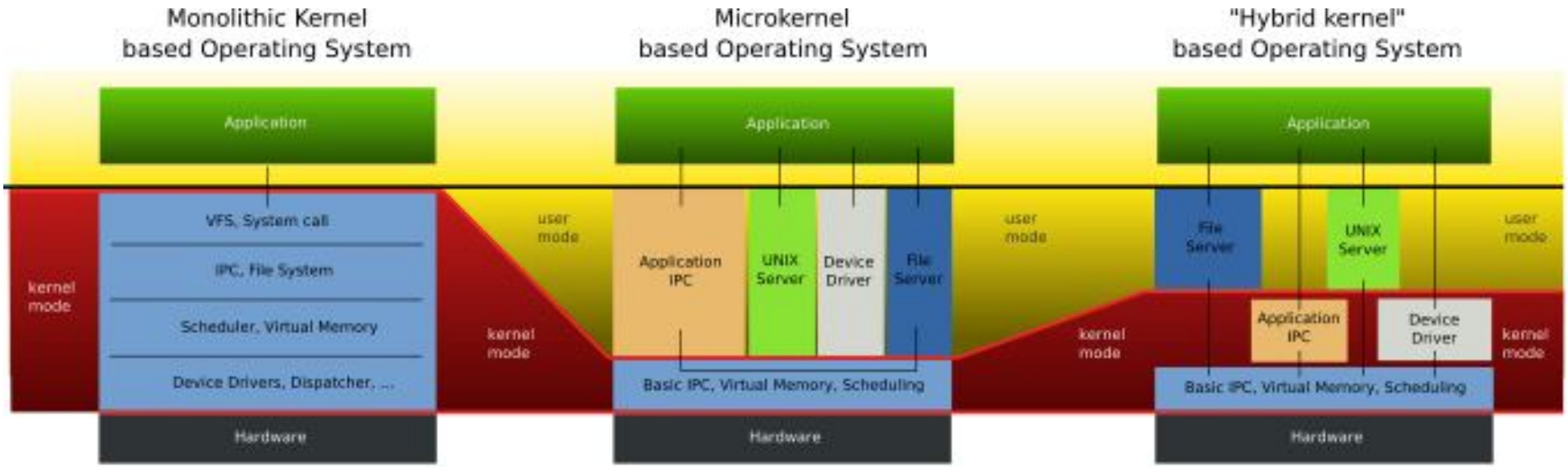
Arquitectura micronúcleo

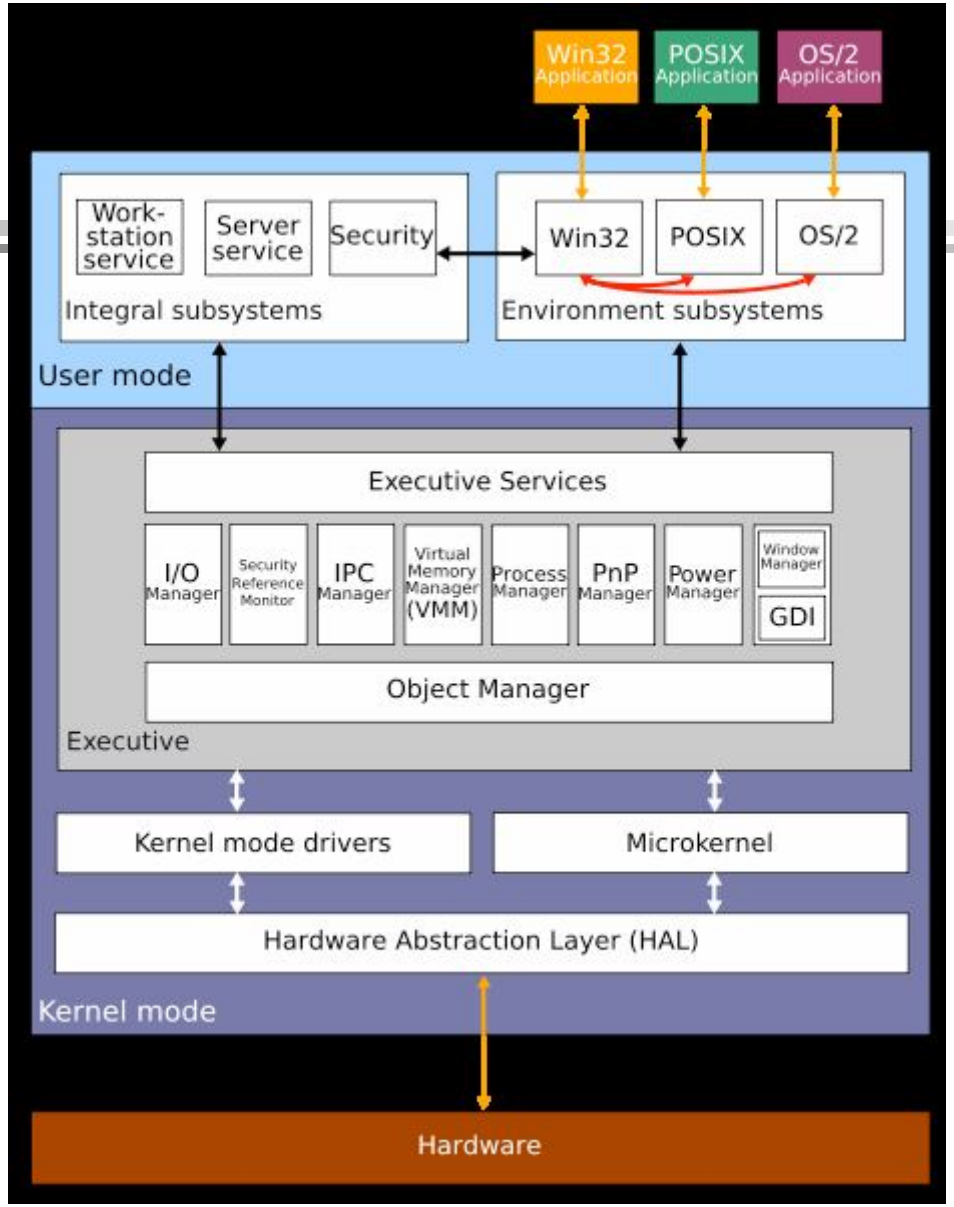


Monolítico versus Micronúcleo

(wikipedia)

Realmente hoy día no se usan modelos puros, sino híbridos





Comunicación IPC
||
Gestión de comunicación entre procesos

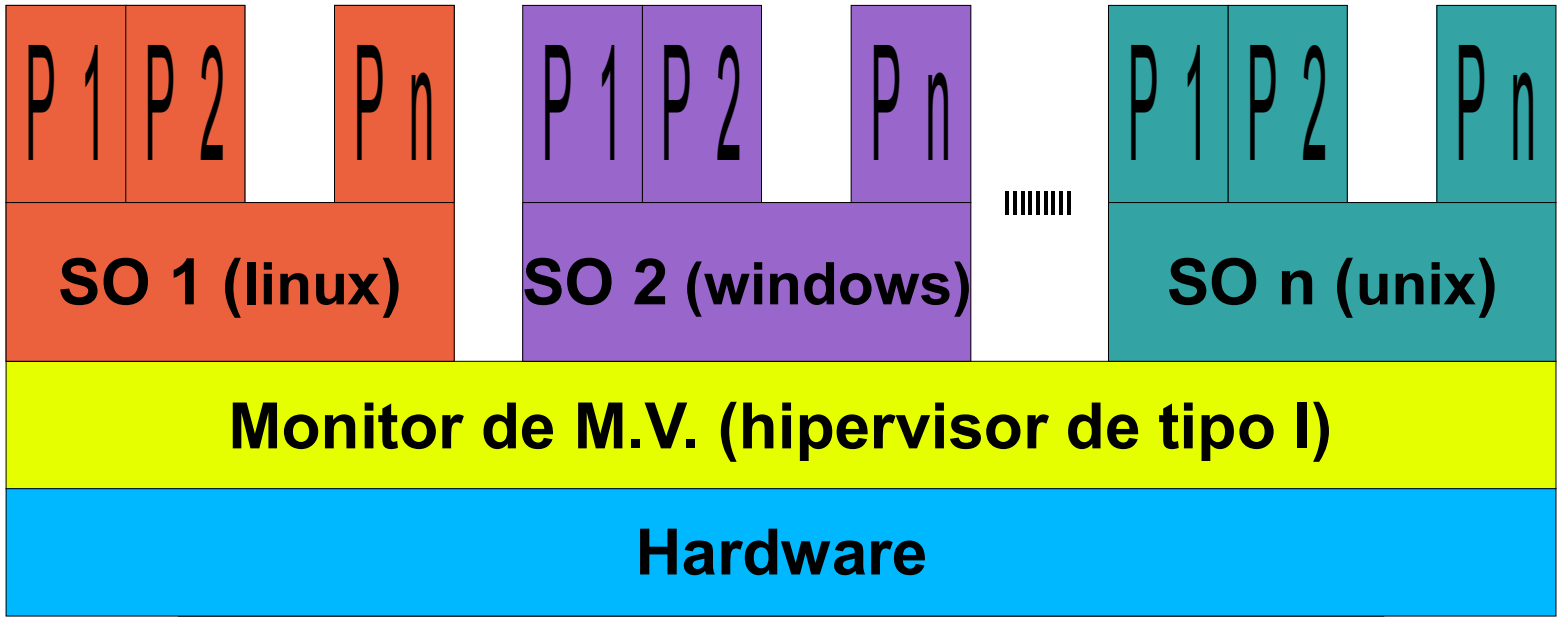
Estructura de Windows 2000 (wikipedia)

Arquitectura: Máquinas Virtuales

- Software que implementa una máquina virtual (= o ≠ máquina real). Cada copia es una réplica exacta del hardware.
- El SO crea una máquina virtual pero extendida (abstracción del hardware)
- Técnica de nuevo en auge actualmente (+ de 40 años)
- La capacidad de procesamiento actual mitiga la ineficiencia de las Máquinas Virtuales
- Los procesadores más actuales incluyen soporte para la misma

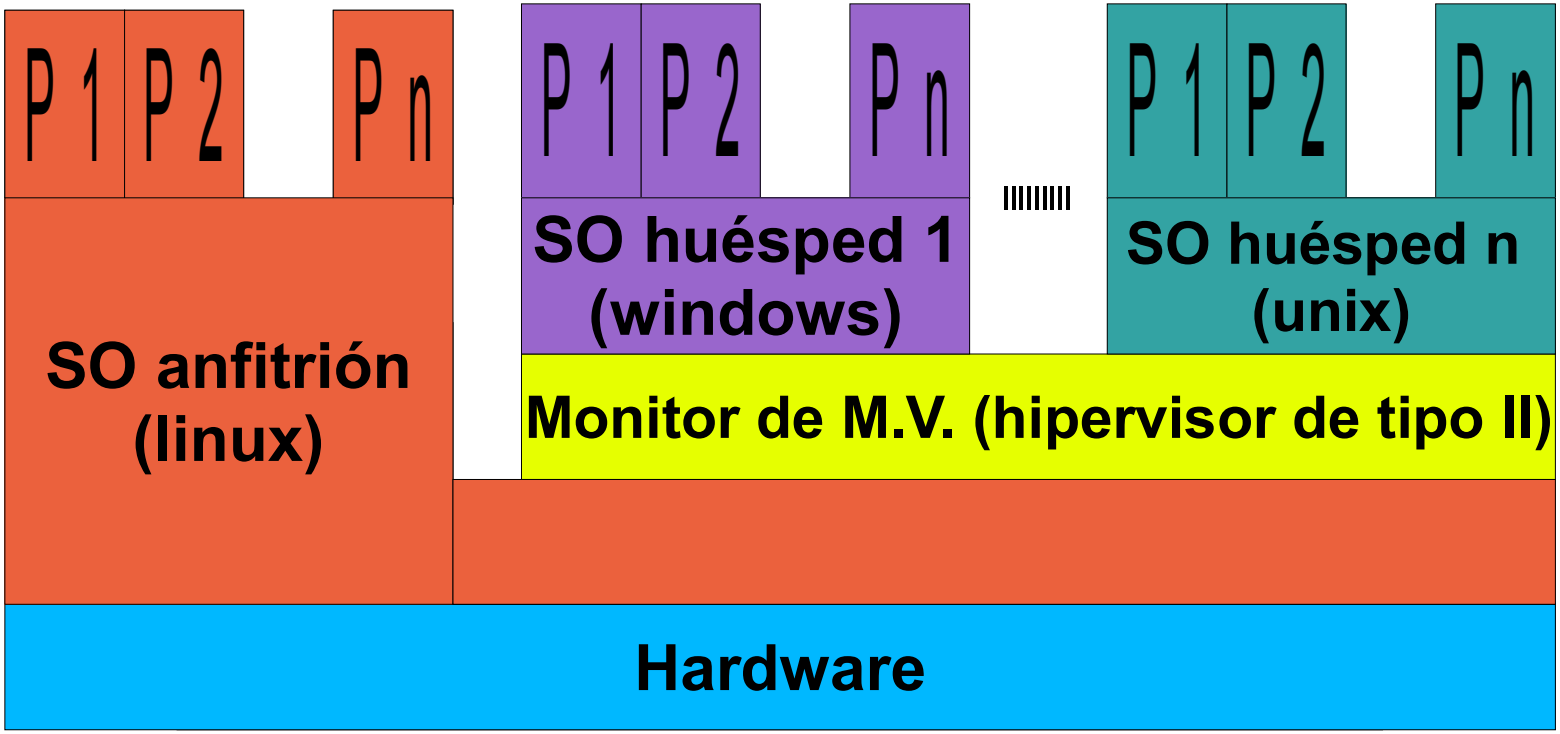
Arquitectura: Máquinas Virtuales (y II)

- Una petición de servicio es atendida por la copia de SO sobre la que se ejecuta (figuras Carretero)



Arquitectura: Máquinas Virtuales (y III)

Las M.V. funcionan como programas. Para que esto funcione el S.O. debe ceder permisos especiales.

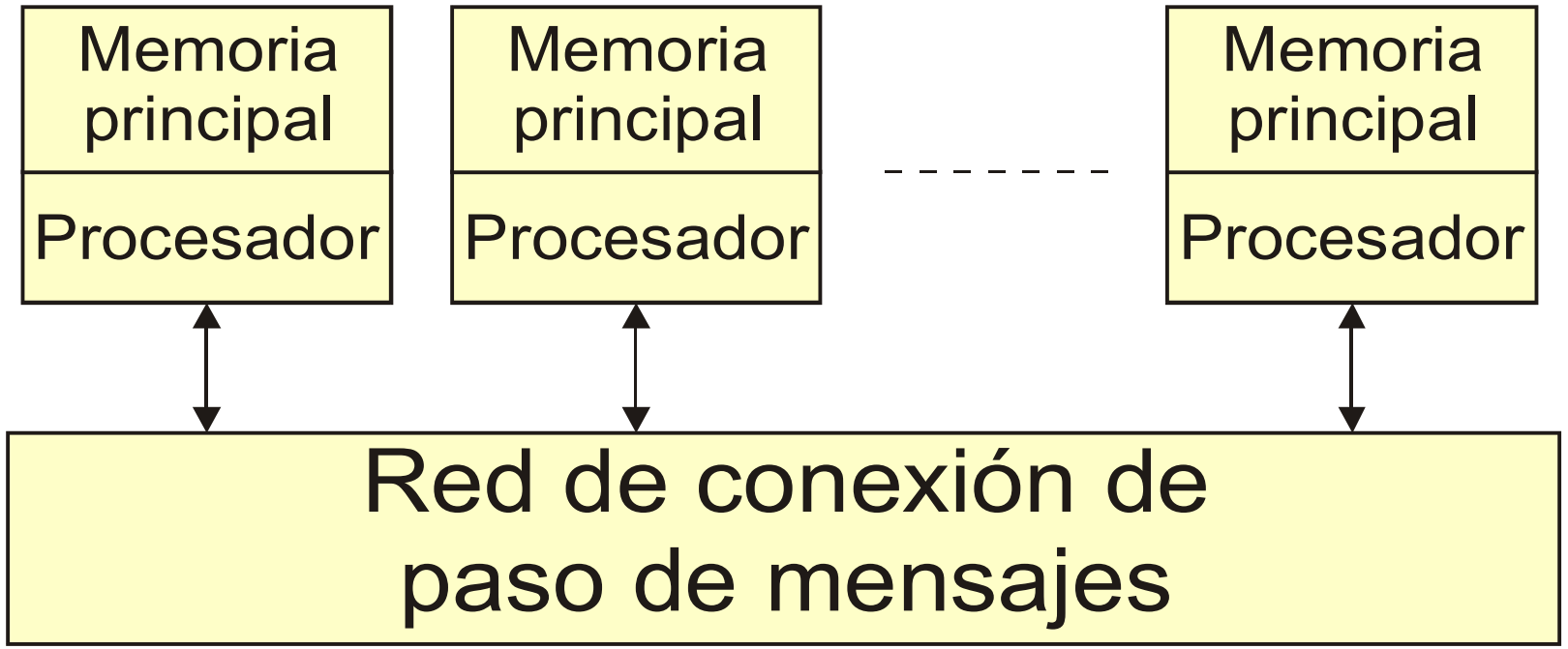


1.2. SO's de propósito específico: SO de Tiempo Real

- Los SOTR se utilizan para aplicaciones especializadas, p.ej. sistemas de control, ...
- Idea básica: SO debe garantizar la respuesta a sucesos físicos en intervalos de tiempo fijos
- Problema: planificar las actividades con el fin de satisfacer todos los requisitos críticos
- Con el uso de aplicaciones de video sobre PC's, todos los SO's tendrán pronto requisitos de tiempo-real

Arquitectura Multicomputador

Es como si tuviésemos varios ordenadores conectados a una red.



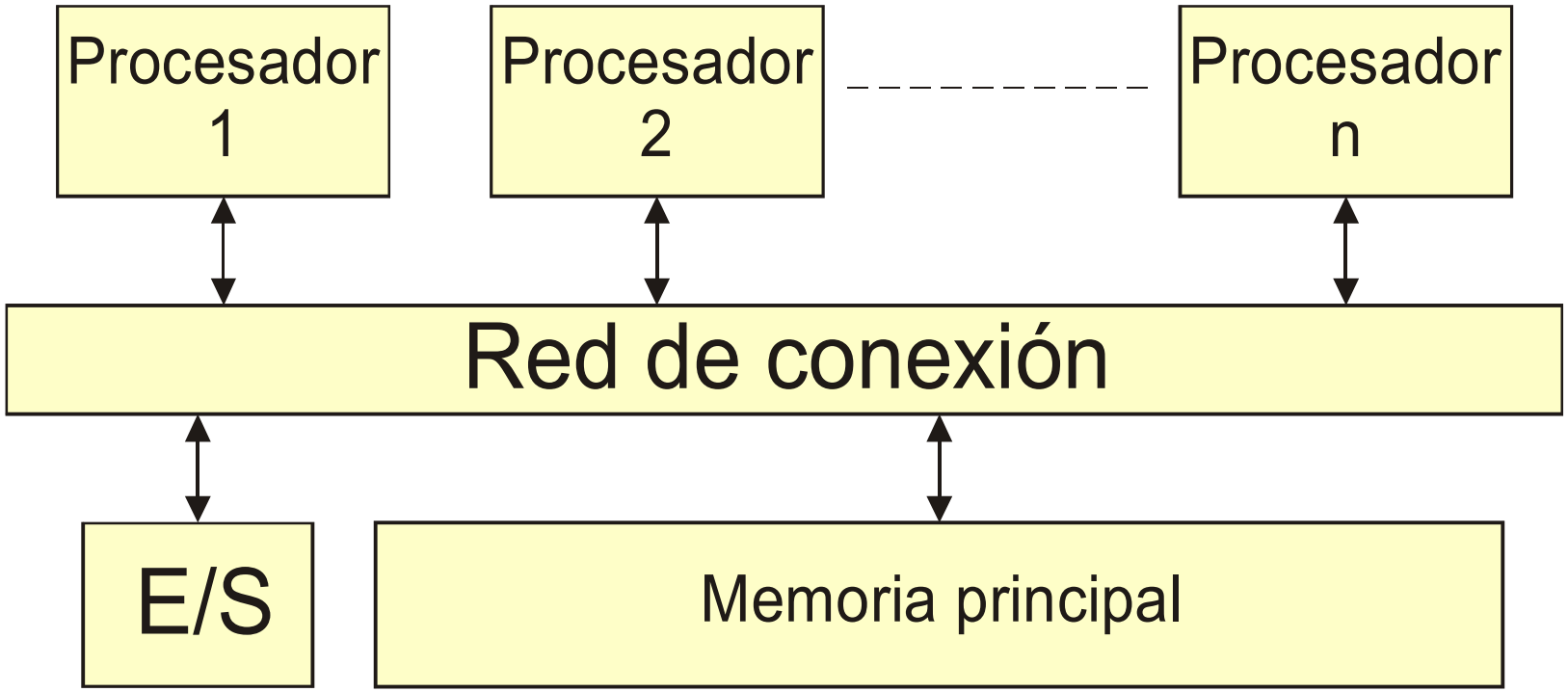
SO's de Red

- Los usuarios son conscientes de la existencia de varias computadoras
- Cada nodo ejecuta su propio SO local y tiene sus propios usuarios
- Lo que les diferencia de los SO de un solo procesador es la necesidad de software especial como:
  - » controlador de interfaz de la red
  - » programas de conexión y acceso a archivos remoto

SO's Distribuidos

- Sistemas débilmente acoplados - sistemas sin memoria común
- Característica fundamental: Transparencia
- Permiten la compartición de recursos distribuidos, hardware o software
- Permiten algún paralelismo, pero el aumento de velocidad no es el objetivo
- Aumentan la fiabilidad del sistema

Arquitectura multiprocesador



SO's Paralelos

- Sistemas multiprocesadores fuertemente acoplados (los procesadores comparten una memoria y el reloj)
- Dos tipos de multiprocesamiento:
  - » Simétrico (SMP) - cada procesador ejecuta una copia idéntica del SO - buen rendimiento
  - » Asimétrico (ASMP) - Un procesador *maestro* ejecuta el SO, los procesadores *esclavos* ejecutan procesos de usuario. Peor escalabilidad

Si deja de funcionar el maestro se jode el sistema.

También puede haber sobrecarga de llamadas.

SO's Paralelos (v II)

- Soportan aplicaciones paralelas que desean obtener aumento de velocidad de tareas computacionalmente complejas
- Necesitan primitivas básicas para dividir una tarea en múltiples actividades paralelas
- Proporciona una comunicación y sincronización eficiente entre esas actividades
- Tolerancia a fallos