Tema 2

ESTRUCTURA SISTEMAS OPERATIVOS

Índice

- Núcleo SO (kernel)
- Mecanismos transferencia al kernel
- Tipos de estructuras de SO
- Casos de estudio:
  - Windows
  - Linux

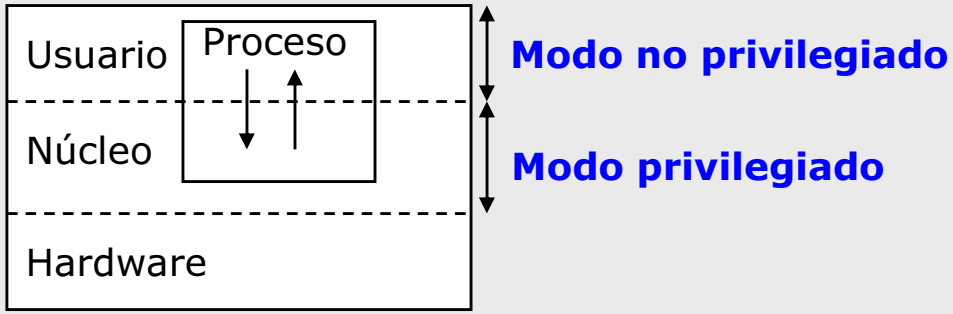
El núcleo (kernel) del S.O.

- Es la parte más importante y crítica del SO y por lo tanto, se encuentra permanentemente cargada en memoria.
- Funciones del núcleo del SO:
  - Gestión de interrupciones
  - Planificación de la CPU
  - Gestión de procesos
  - Sincronización y comunicación entre procesos
  - …
- El núcleo es un área de ejecución privilegiada (se puede hacer cualquier cosa), y por lo tanto requiere protecciones fuertes.

Modo dual de trabajo

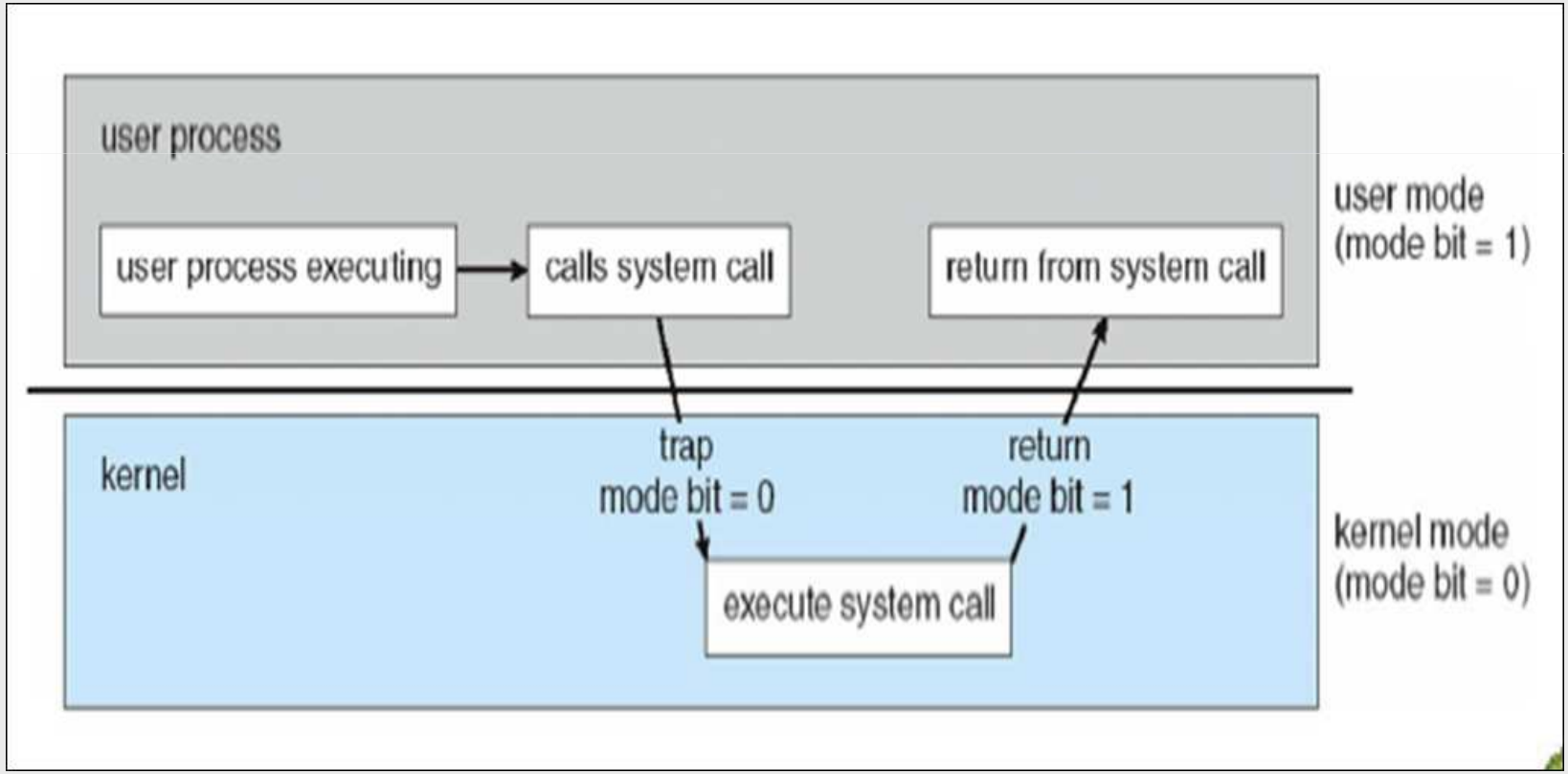
- Mecanismo de protección hardware que permite proteger los recursos de la máquina y asegurar que un programa no afectará a otro
- Permite, al menos, 2 modos de ejecución: usuario (1) y sistema (0).
- Instrucciones privilegiadas sólo posibles en modo sistema.
  - E/S:
    - siempre en modo sistema
  - Memoria:
    - registros base y límite por programa
    - accesos fuera originan un fallo de protección

Acceso al núcleo (kernel) del S.O.



- El cambio de modo de ejecución implica también un cambio contexto desde el modo usuario al modo kernel.
- El cambio de contexto implica:
  - Modificar espacio de direccionamiento
  - Modificar el modo de ejecución de modo no privilegiado a modo privilegiado ó viceversa.
- Mecanismos de entrada al núcleo:
  - Traps,
  - Interrupciones
  - Excepciones.

Cambio modo ejecución



Interrupciones

- Son sucesos asíncronos independientes al proceso en ejecución.
- Proceso de tratamiento interrupción:
  - Almacenar contexto proceso y cambiar a modo protegido.
  - Determinar la causa de la interrupción.
  - Buscar la dirección de la RTI
  - Llamar a la RTI.
  - Restaurar contexto del proceso y volver a modo usuario.
- Prioridad interrupciones.

Excepciones

- Son roturas de secuencia no previstas, provocadas por la ejecución de usuario en curso (normalmente errores).
- Tratamiento:
  - Tratar la excepción por parte del SO.
  - Si no se soluciona el problema:
    - Enviar señal al proceso.
    - Proceso actúa en consecuencia:
      - Ejecuta el gestor especificado por el proceso.
      - Ejecuta el gestor por defecto.

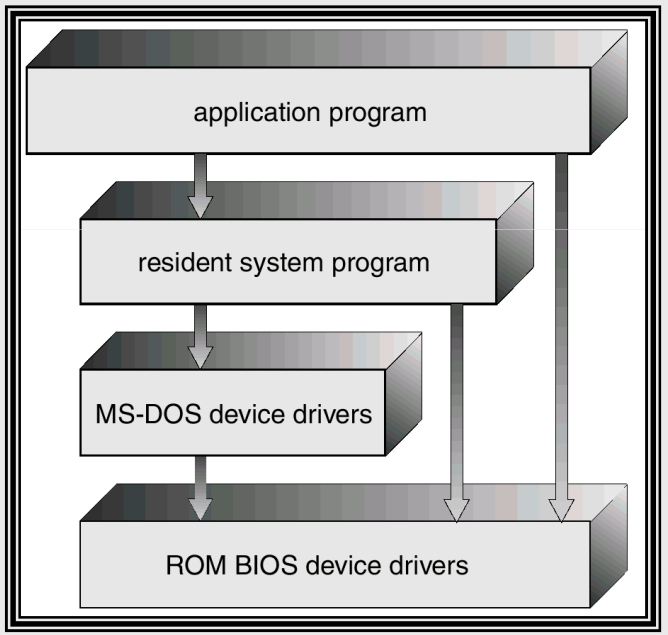
Estructura S.O.

- Atendiendo a la estructura interna del núcleo, los S.O. se pueden clasificar:
  - S.O. monolíticos.
  - S.O. por capas ó niveles.
  - S.O. basados en micro-kernel
  - S.O. basados en máquinas virtuales

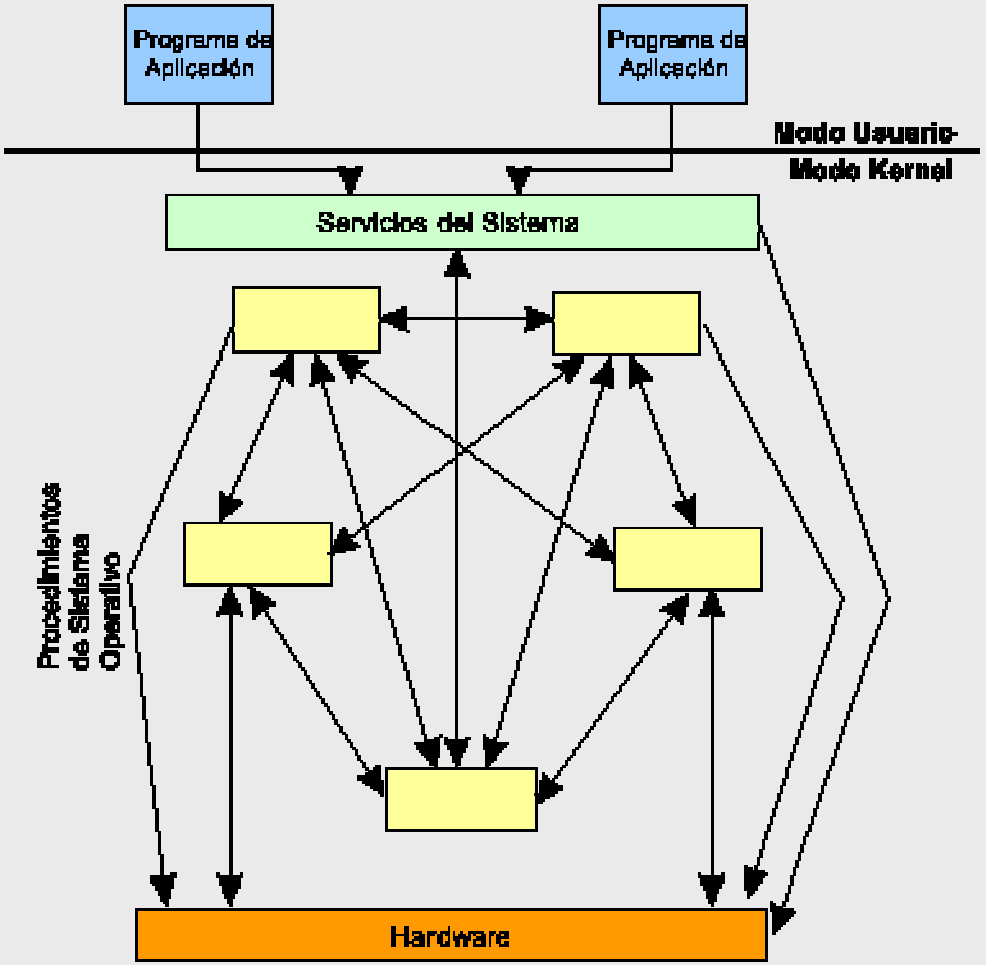
S.O. monolíticos.

- S.O. monolíticos.
  - No tienen una estructura bien definida
  - El s.o. está descrito como un conjunto de procedimientos que se pueden llamar entre si, sin ninguna limitación.
  - S.O complejos y difíciles de implementar y depurar
- Ejemplo: MS-DOS

Estructura MS-DOS



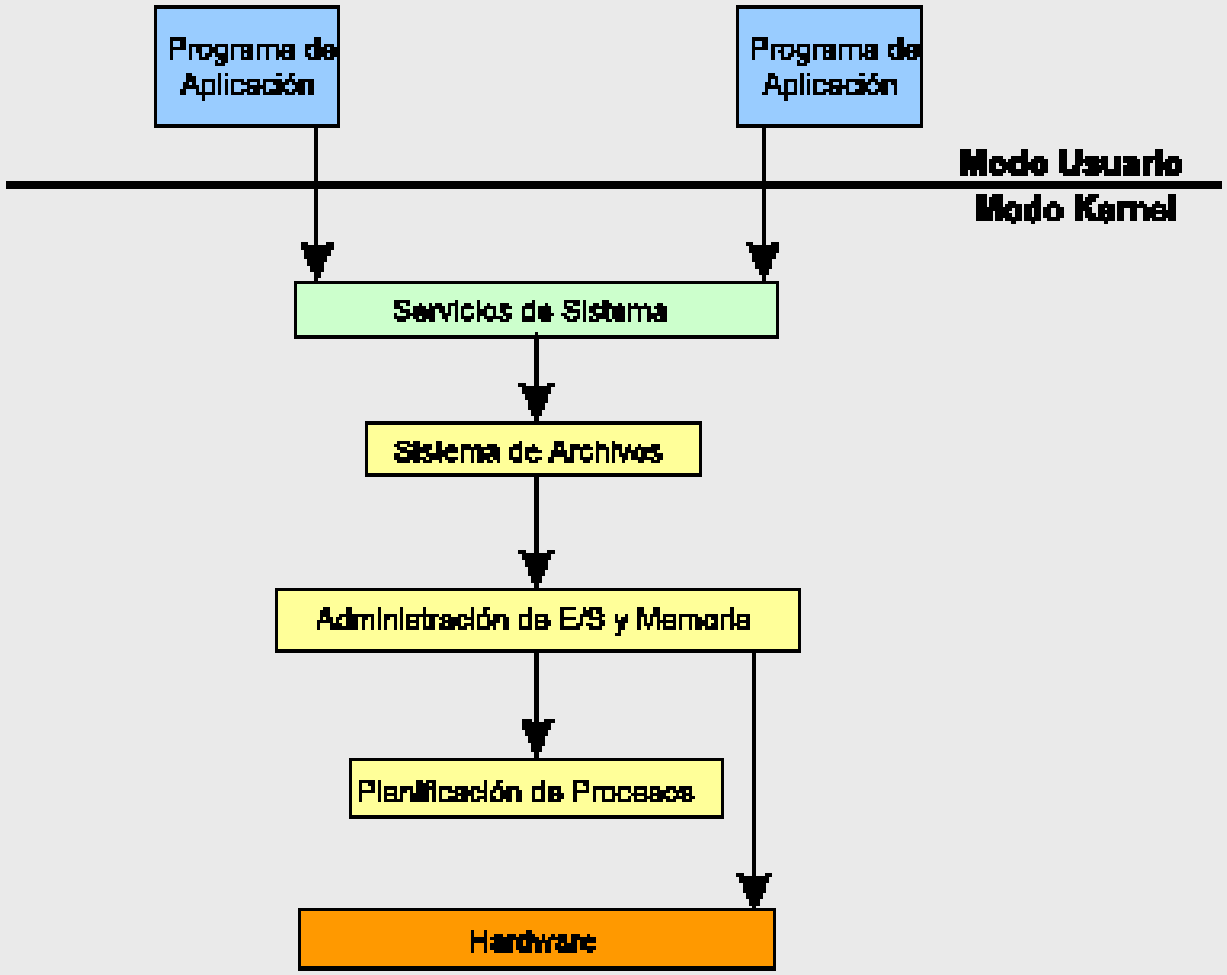
Estructura SO Monolíticos



S.O. por capas ó niveles.

- Existen distintos niveles ó capas de sistema, las cuales van creando diferentes niveles de abstracción.
- La capa inferior es el hardware, mientras que la capa superior es el interfaz con el usuario.
- Una capa solo se puede comunicar con las capas contiguas.
- Ventajas:
  - Independencia entre las capas.
  - Permite describir el SO de forma más clara.
- Desventajas:
  - Rendimiento.
  - Dificultad definir apropiadamente las capas debido a a las limitaciones de comunicación entre capas.

Estructura SO por Capas



Estructura Microkernel

- Mueve en lo posible la funcionalidad desde el kernel hacia el espacio de usuario.
- La comunicación se produce entre los módulos de usuario utilizando paso de mensajes.
- Ventajas:

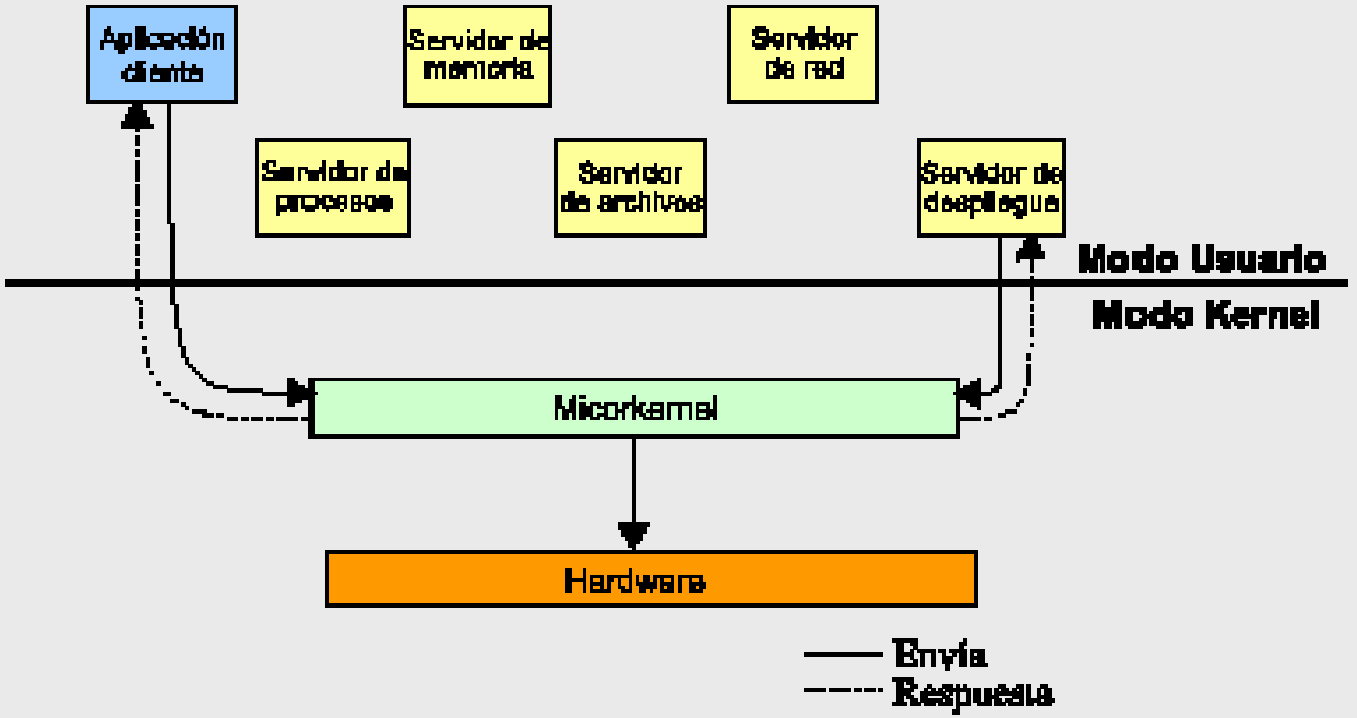
  - Fácil de extender el microkernel

  - Fácil de portar el SO hacia nuevas arquitecturas hardware

  - Más confiable (una menor porción del código se está ejecutando en modo kernel)

  - Más seguro

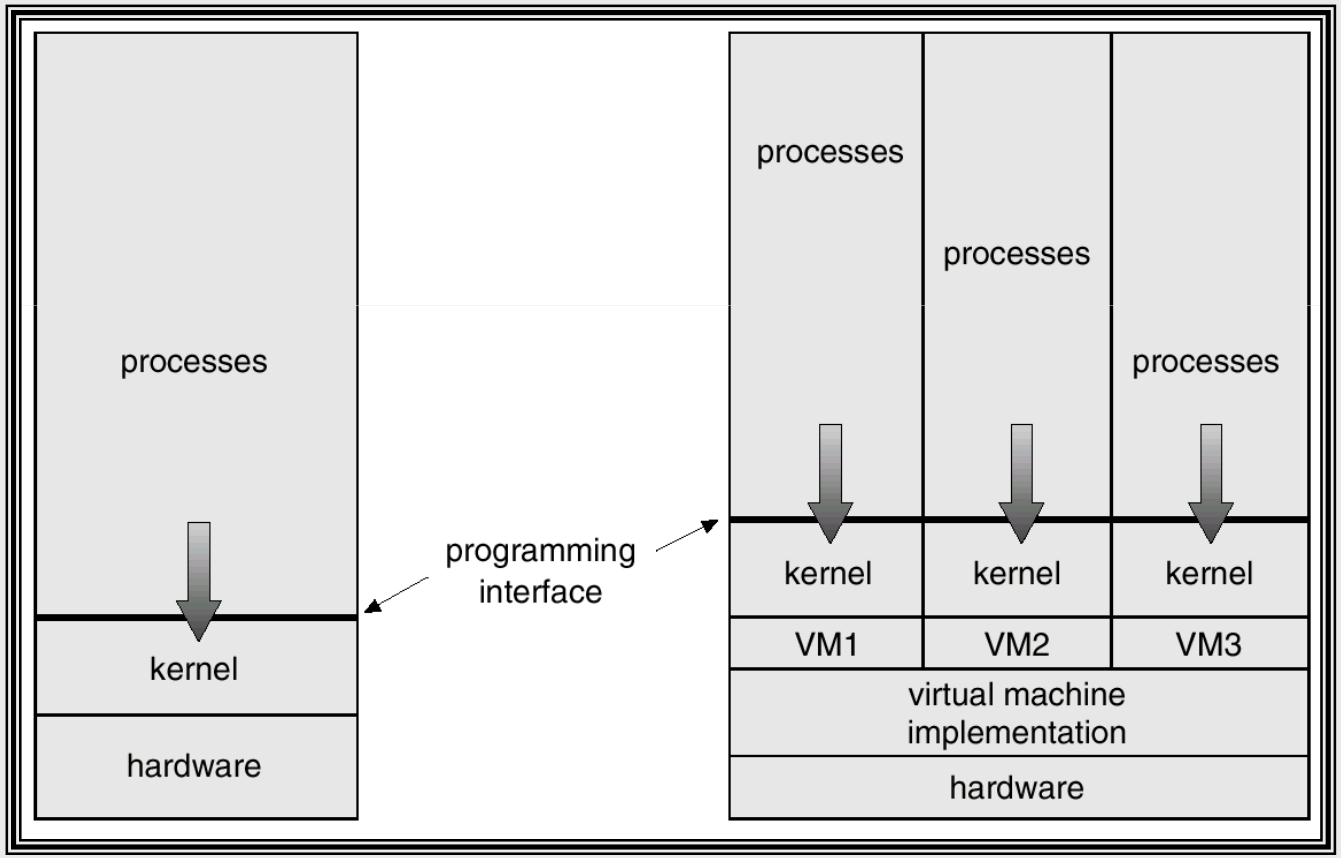
Estructura SO microkernel



Máquinas Virtuales

- El SO crea la ilusión de múltiples procesos, cada uno de ellos ejecutándose en su propio procesador (máquina virtual).
- Una MV crea una copia idéntica del hardware
  - Procesador con su propia memoria y E/S
  - Sobre una MV se puede ejecutar cualquier SO
  - Sobre el mismo computador varios SSOO a la vez
- Los recursos del computador físico están compartidos para crear las máquinas virtuales.
  - Planificación UCP para MV ➔ multiproceso
  - Spooling y sistema ficheros ➔ multiplexación E/S
- Ventajas:
  - Protección sencilla, muy modular, bueno para investigación y desarrollo ya que permite ejecutar diferentes SO
- Inconvenientes:
  - Difícil compartir recursos
  - Difícil implementar duplicados exactos del HW
- Ej. VM-370, MS-DOS en Windows, **VMware**

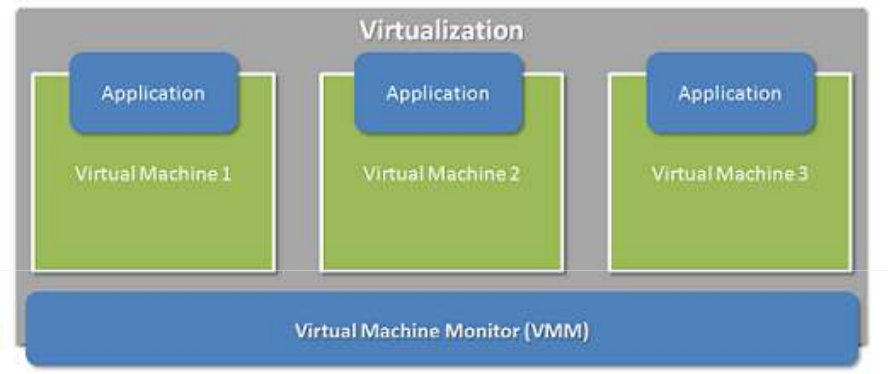
Estructura SO basado en Máquinas Virtuales



Sin Máquina Virtual

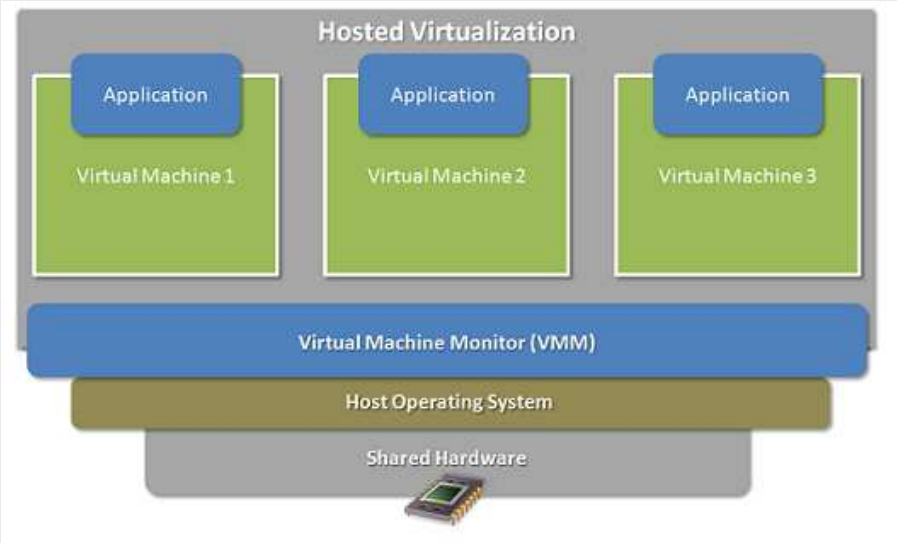
Maquina Virtual

Tipos de Virtualización



- Máquina virtual o hipervisor de tipo 1
  - Se ejecutan encima del hardware y los comparten entre las diferentes VM
  - Requieren de soporte específico del HW

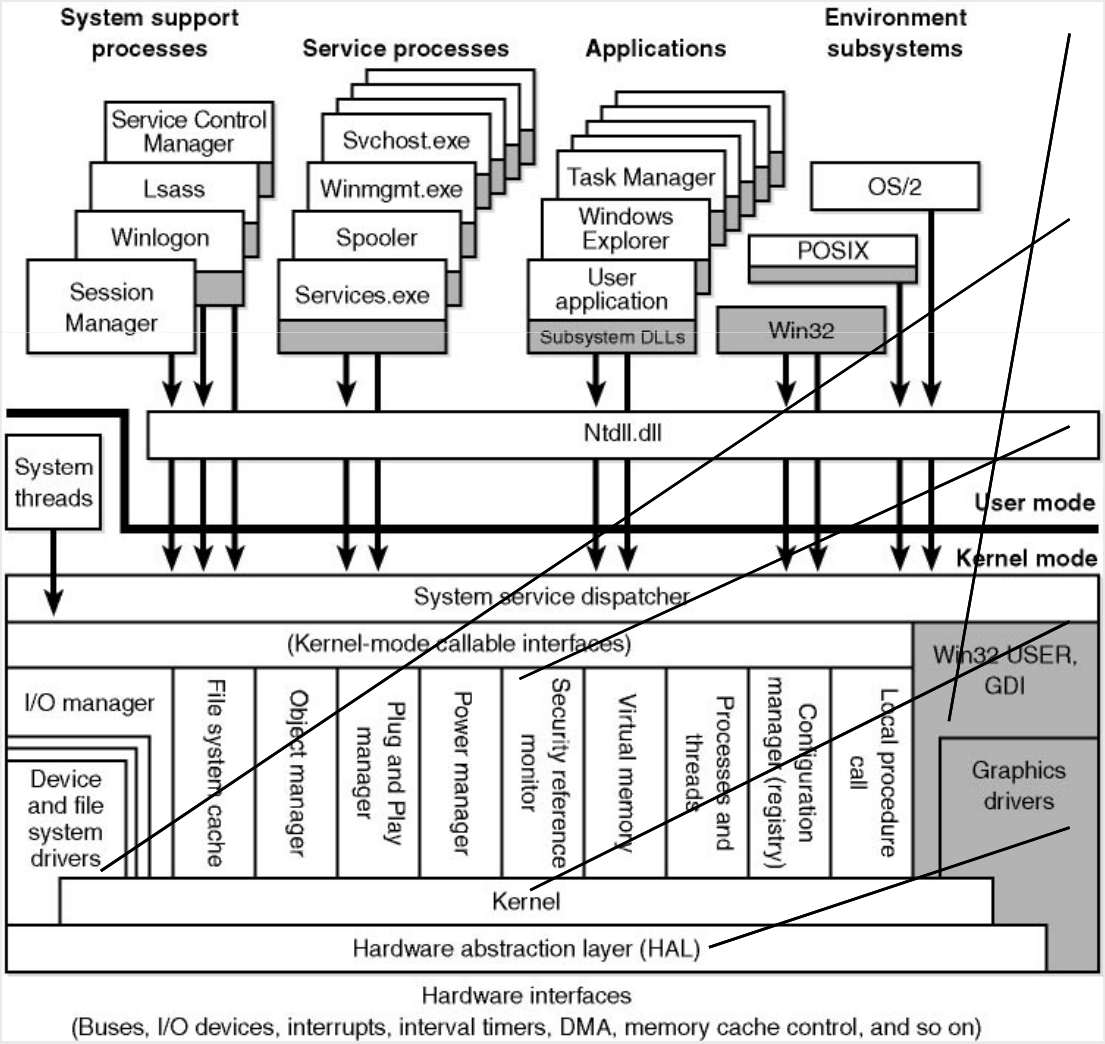
- Máquina virtual o hipervisor de tipo 2
  - Se ejecutan dentro de un sistema operativo Huesped, creando la máquina virtual a partir de los recursos gestionados por este
  - No requieren soporte específico del HW



Caso de estudio: Windows

- Características (NT, 2000, XP, vista,…):
  - Arquitectura Microkernel (modelo cliente-servidor a nivel usuario).
  - Diseño Orientado a Objetos.
  - Soporte Multihilos.
  - Soporte Multiprocesamiento Simétrico.
  - Compatibilidad otros sistemas operativos (subsistema de ambiente)
  - Abstracción Hardware (HAL)

Windows: Arquitectura



Sistema gráfico y de ventanas: Implementa la interfaz gráfica de usuario (GUI)

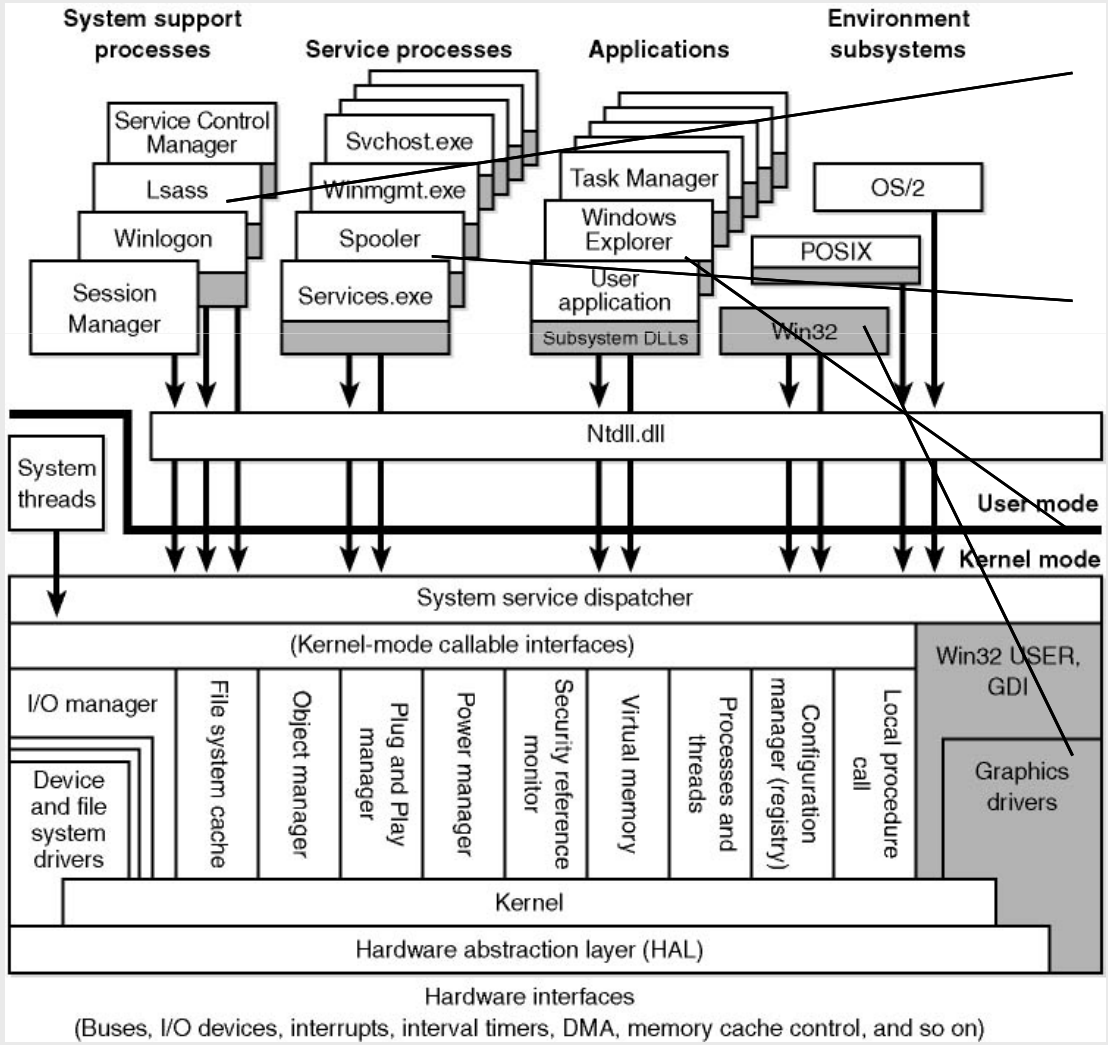
Drivers de dispositivos: Conjunto de librería dinámicas y dirvers que permiten traducir las peticiones de E/S del usuario a peticiones a los dispositivitos.

Ejecutor: Contiene los servicios básicos del SO (Gestión de memoria, E/S, procesos y threads)

Kernel: Controla la ejecución del procesador/es (planificación hilos, cambio contexto, gestión interrupciones)

HAL (Capa de abstracción del hardware): Permite aislar el kernel del S0 de las características especificas del hardware.

Windows: Arquitectura



Procesos soporte sistema: Procesos en modo usuario necesarios para gestionar el sistema (gestor de sesión, subsistema de autentificación)

Procesos de servicio: Servicios utilizados por los desarrolladores para extender la funcionalidad del sistema (ejecución en background)

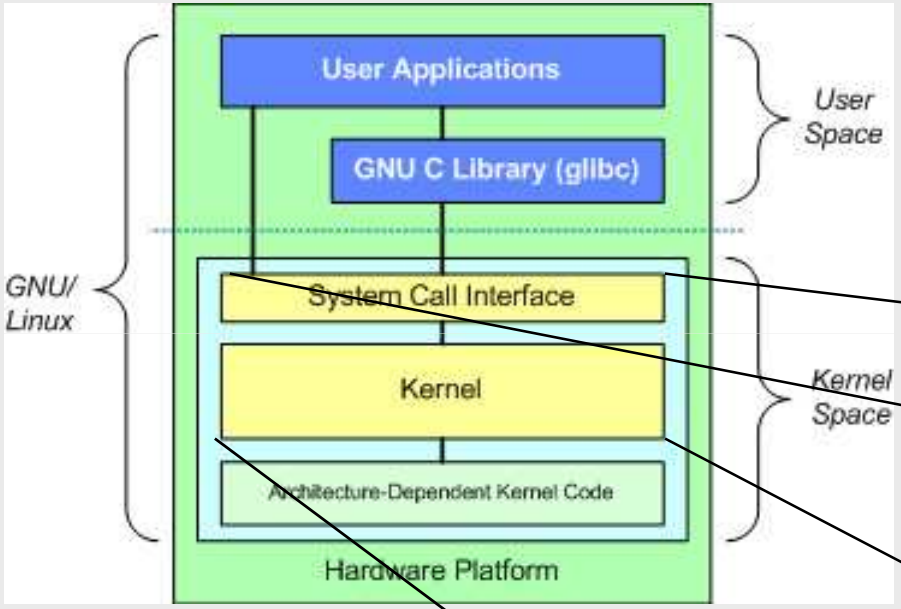
Aplicaciones de usuario: Ejecutables y DLLs que proporcionan la funcionalidad final al usuario.

Subsistemas de ambiente protegido: Proporciona diferentes ambientes de ejecución compatibles con las APIS de programación de diferentes sistemas operativos (Windows, POSIX, OS/2)

Caso de estudio: Linux

- Características:
  - Arquitectura monolítica híbrida basada en módulos:
    - Enlazado dinámico
    - Módulos apilables
  - Basado en procesos pero con soporte a hilos.
  - Soporte Multiprocesamiento Simétrico.
  - Abstracción Hardware

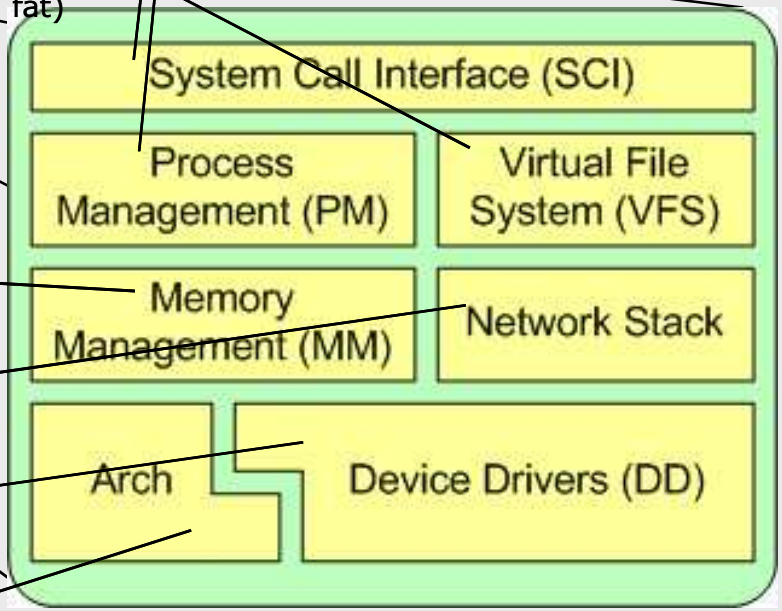
Linux: Arquitectura



Interfase de llamadas al sistema: Proporciona los mecanismos para realizar llamadas desde el espacio de usuario hacia el kernel del SO

Gestión de procesos: Se encarga de la gestión y planificación de los procesos y sus hilos de ejecución

Sistema de ficheros virtual: Proporciona una interfase común para la abstracción de sistemas de archivos, soportando múltiples SA (Ext2-3, reiser, fat)



Gestión de memoria: Soporta la gestión de la memoria física y la memoria virtual.

Pila de protocolos de red: Soporte para TCP/IP y UDP.

Drivers dispositivos: Gestión de drivers dispositivos y su integración con el núcleo.

Código dependiente de la arquictectura