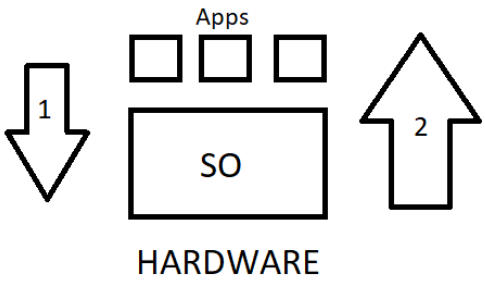
Resumen Tema 1 "Introducción"

Sistema Operativo: Es un programa o Software (necesita procesador y memoria para ejecutarse).

Existen dos pespectivas:



1. De arriba hacia abajo:
    a. Los programas de aplicaciones son los "clientes" del S.O.
    b. El SO "oculta" el HW y presenta a los programas abstracciones más simples de manejar.
    c. Abstracción con respecto a la arquitectura (conjunto de instrucciones, E/S, estructura del bus)
    d. Comparación: uso de escritorio y uso de comandos de texto.
2. De abajo hacia arriba
    a. Visión del SO como un administrador de recursos.
    b. Hay un Software diseñado, estructurado para administrar los recursos de HW (no todos los SO tienen el mismo objetivo).
    c. Administra los recursos de HW de uno o más **procesos** (*abstracción que utilizan los SO para utilizar la CPU*).
    d. Provee un conjunto de servicios a los usuarios del *sistema*
    e. Maneja la memoria secundaria y dispositivos de I/O.
    f. Ejecución simultánea de procesos
    g. Multiplexación en tiempo (CPU) y en espacio (memoria)
    h. Archivo: abstraccion que me permite usar el recurso de almacenamiento.

El Sistema Operativo:

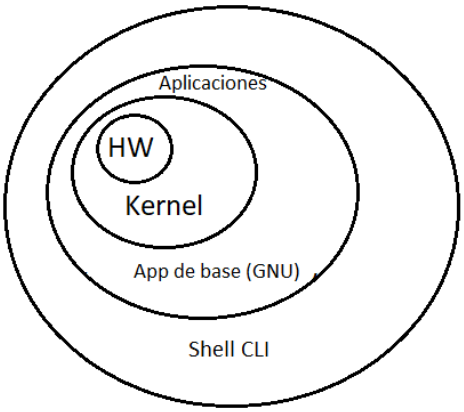
- Gestiona el HW
- Controla la ejecución de los procesos
- Interfaz entre aplicaciones y HW
- Actúa como intermediario entre un usuario de una computadora y el HW de la misma

Objetivos:

- Comodidad → Hacer más fácil el uso del hardware (PC, servidor, switch, router, controlador específico)
- Eficiencia → Hacer un uso más eficiente de los recursos del sistema
- Evolución → Permitir la introducción de nuevas funciones al sistema sin interferir con funciones anteriores

Componentes:

- Kernel o núcleo.
- Shell(cmd) → GUI/CUI o CLI
    - Herramienta que utiliza el usuario para mandarle comandos al SO. Línea de comando que traduce la operación al kernel.
- Herramientas
    - Editores, compiladores, librerías, etc.



Kernel: De todo el SO. Es una porción de código que siempre esta en la memoria principal por estar encargado de la administración de los recursos.

Implementa servicios esenciales:

- Manejo de memoria
- Manejo de la CPU
- Administración de procesos

- Comunicación y Concurrencia
- Gestión de E/S

Servicios del SO:

- Administración y planificación del procesador
    - Multiplexación de la carga de trabajo
    - Imparcialidad, "justicia" en la ejecución (Fairness)
    - Que no haya bloqueos
    - Manejo de Prioridades
- Administración de memoria
    - Administración de memoria eficientemente
    - Memoria física vs memoria virtual. Jerarquías de memoria
    - Protección de programas que compiten o se ejecutan concurrentemente
- Administración del almacenamiento del sistema de archivos
    - Acceso a medios de almacenamiento externos
- Administración de dispositivos
    - Ocultamiento de dependencias de HW
    - Administración de accesos simultáneos
- Detección de errores y respuestas
    - Errores de HW internos y externos
        - Errores de Memoria/CPU
        - Errores de dispositivos
    - Errores de SW
        - Errores aritméticos
        - Acceso no permitido a direcciones de memoria
    - Incapacidad del SO para conceder una solicitud de una aplicación
- Interacción del usuario (Shell)
- Contabilidad sobre los recursos que se están utilizando
    - Recoger estadísticas del uso
    - Monitorear parámetros de rendimiento
    - Anticipar necesidades de mejoras futuras
    - Dar elementos si es necesario facturar tiempo de procesamiento.

Problemas que el Kernel debe evitar:

Que un *proceso* se apropie de la CPU

Se necesita apoyo del HW (interrupciones por clk que garantiza que un proceso no se apropie de la CPU) manipulado por el Kernel por problemas, como por ej, que un proceso se apropie de la CPU.

Las aplicaciones de usuario tienen que tener restricciones de lo que se puede hacer.

Funciones principales de un SO:

- Brindar abstracciones de alto nivel a los procesos de usuario
- Administrar eficientemente el uso de la CPU
- Administrar eficientemente el uso de la memoria
- Brindar asistencia al proceso de E/S por parte de los procesos

Problemas que un SO debe evitar

- Que un proceso se apropie de la CPU
- Que un proceso intente ejecutar instrucciones privilegiadas (Ej: E/S)
- Que un proceso intente acceder a una posición de memoria fuera de su espacio permitido.

Entonces, el SO debe:

- Gestionar/controlar el uso de la CPU
- Detectar intentos de ejecución de instrucciones privilegiadas
- Detectar accesos ilegales a memoria
- Proteger el vector de interrupciones –Así como las RAI (Rutinas de atención de interrupciones)

Apoyo del Hardware:

- **Modos de Ejecución**: Define limitaciones en el conjunto de instrucciones que se puede ejecutar en cada modo
- **Interrupción de Clock**: Se debe evitar que un proceso se apropie de la CPU
- **Protección de la Memoria:** Se deben definir límites de memoria a los que puede acceder cada proceso (registros base y límite)

Modos de ejecución

- El bit en la CPU indica el modo actual
- Las instrucciones privilegiadas deben ejecutase en modo **Supervisor o Kernel**.
    - Necesitan acceder a estructuras del Kernel, o ejecutar código que no es del proceso.
- En modo **Usuario**, el proceso puede acceder solo a su espacio de direcciones, es decir a las direcciones *propias.*
- El Kernel del SO se ejecuta en modo supervisor.
- El resto del SO y los programas de usuario en modo usuario (subconjunto en instrucciones permitidas).

- Cuando se arranque el sistema, arranca con el bit en modo supervisor.
- Cada vez que comienza a ejecutarse un proceso de usuario, este bit se DEBE PONER en modo usuario, mediante una Instrucción especial.
- Cuando hay un trap o una interrupción, el bit de modo se pone en modo Kernel. Es la única forma de pasar a modo Kernel y no es el proceso de usuario quien hace el cambio explícitamente.

Cuando el proceso de usuario intenta por sí mismo ejecutar instrucciones que pueden causar problemas (las llamadas instrucciones privilegiadas), el HW lo detecta como una operación ilegal y produce un trap al SO

En Windows en WIN2000 el modo núcleo ejecuta los servicios ejecutivos. El modo usuario ejecuta los procesos de usuario.

Cuando un programa se bloquea en modo usuario, a lo sumo se escribe un suceso en el registro de sucesos. Si el bloqueo se produce estando en modo supervisor se genera la BSOD (pantalla azul de la muerte).

- El procesador Intel 8088 no tenía modo dual de operación ni protección por hardware.
- En MS-DOS las aplicaciones pueden acceder directamente a las funciones básicas de E/S para escribir directamente en pantalla o en disco.

Modo Kernel:

- Gestión de procesos: Creación y terminación, planificación, intercambio, sincronización y soporte para la comunicación entre procesos.
- Gestión de memoria: Reserva de espacio de direcciones para los procesos, Swapping, Gestión y páginas de segmentos
- Gestión E/S: Gestión de buffers, reserva de canales de E/S y de dispositivos de los procesos
- Funciones de soporte: Gestión de interrupciones, auditoría, monitoreo

Modo Usuario:

- Debug de procesos, definición de protocolos de comunicación gestión de aplicaciones (compilador, editor, aplicaciones de usuario)
- En este modo se llevan a cabo todas las tareas que no requieran accesos privilegiados
- En este modo no se puede interactuar con el hardware

- El proceso trabaja en su propio espacio de direcciones

<u>Protección de la memoria</u>

Se basa en delimitar el espacio de direcciones del proceso. Poner límites a las direcciones que puede utilizar un proceso

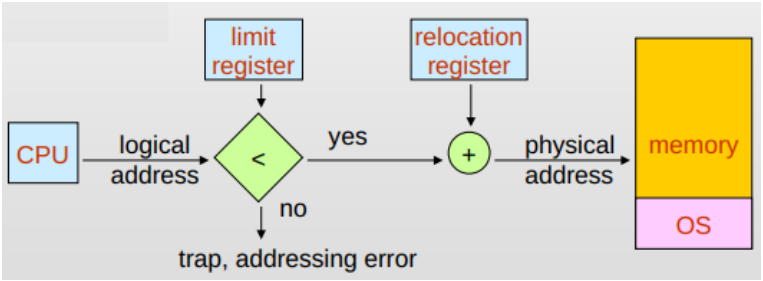
Por ejemplo, Uso de un registro base y un registro límite. El Kernel carga estos registros por medio de instrucciones privilegiadas. Esta acción sólo puede realizarse en modo Kernel

La MP (memoria principal) aloja al SO y a los procesos de usuarios.

El Kernel debe proteger para que los procesos de usuario no puedan acceder donde no les corresponde

El Kernel debe proteger el espacio de direcciones de un proceso del acceso de otros procesos.

Direcciones lógicas: son direcciones que usan los procesos.



Direcciones físicas: son direcciones de memoria RAM.

Protección de la E/S: Las instrucciones de E/S se definen como privilegiadas.

Deben ejecutarse en Modo Kernel

- Se deberían gestionar en el kernel del SO.
- Los procesos de usuario realizan E/S a través de llamadas al SO (es un servicio del SO).

Protección de la CPU: Uso de interrupción por clock para evitar que un proceso se apropie de la CPU.

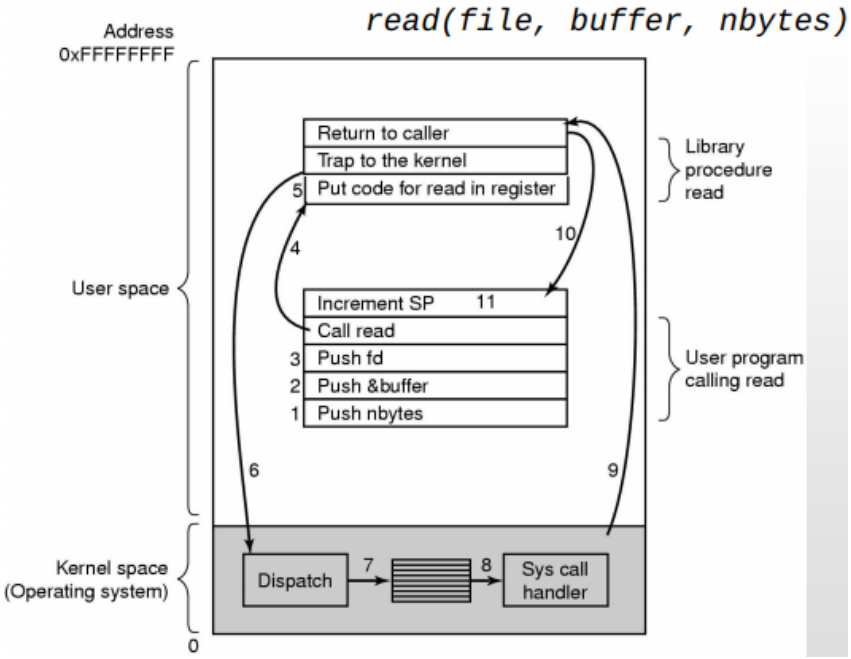
Se implementa normalmente a través de un clock y un contador. El kernel le da valor al contador que se decrementa con cada tick de reloj y al llegar a cero puede expulsar al proceso para ejecutar otro.

- Las instrucciones que modifican el funcionamiento del reloj son privilegiadas.
- Se le asigna al contador el valor que se quiere que se ejecute un proceso.
- Se la usa también para el cálculo de la hora actual, basándose en cantidad de interrupciones ocurridas cada tanto tiempo y desde una fecha y hora determinada.

<u>System Calls:</u> Es la forma en que los programas de usuarios acceden a los servicios del SO.

Los parámetros asociados a las llamadas pueden pasarse de varias maneras: por registros, bloques o tablas en memoria ó la pila. `count=read(file, buffer, nbytes);`

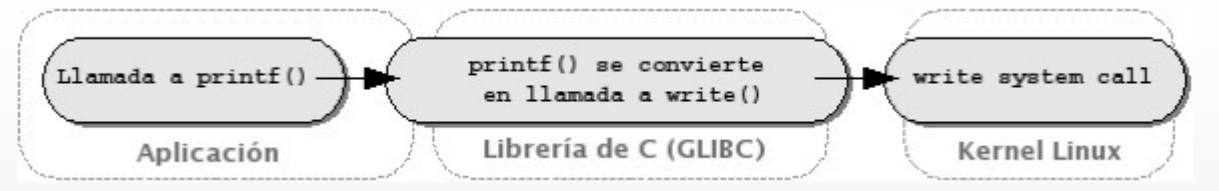
Se ejecutan en *modo kernel* o *supervisor*.



Categorías de System Calls:

- ✓ Control de Procesos
- ✓ Manejo de archivos
- ✓ Manejo de dispositivos
- ✓ Mantenimiento de información del sistema
- ✓ Comunicaciones

Ejemplo – System Call Linux



Para activar iniciar la system call se indica:

- el número de syscall que se quiere ejecutar
- los parámetros de esa syscall

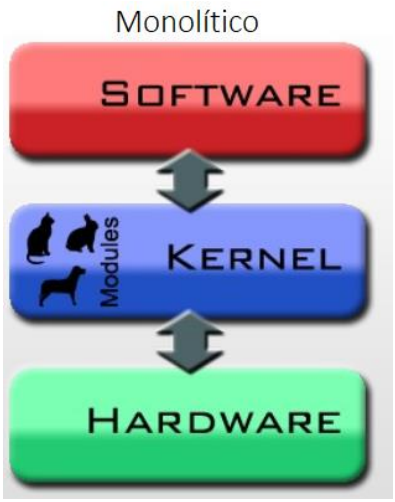
Luego se emite una interrupción para pasar a modo Kernel y gestionar la systemcall

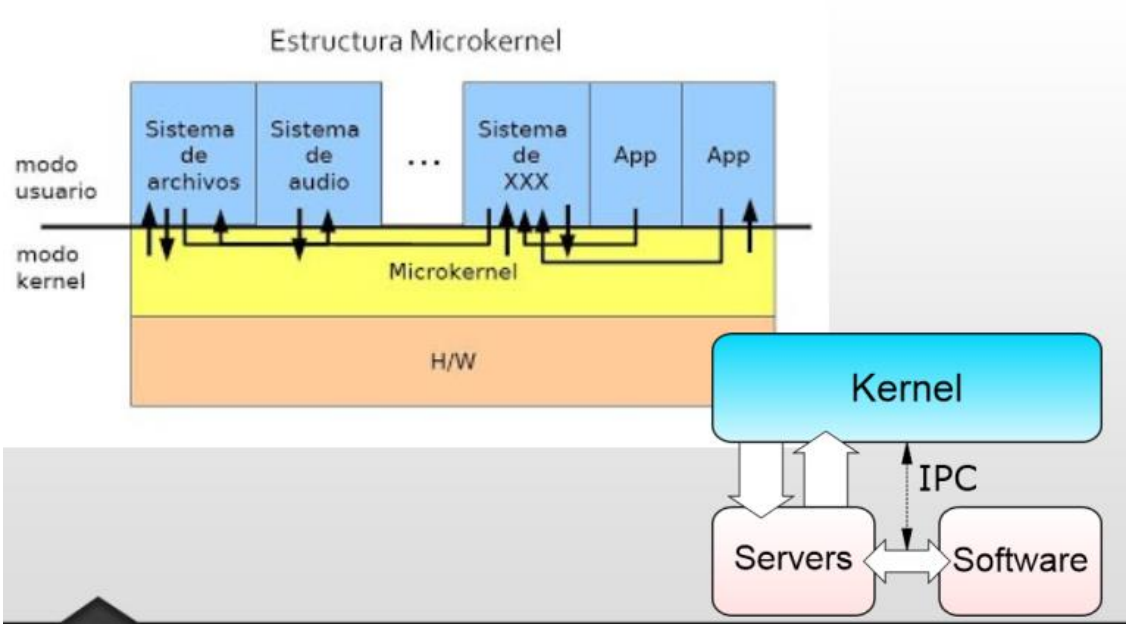
El manejador de interrupciones (syscall handler) evalúa la system call deseada y la ejecuta

Otros ejemplos

UNIX	Win32	Description
fork	CreateProcess	Create a new process
waitpid	WaitForSingleObject	Can wait for a process to exit
execve	(none)	CreateProcess = fork + execve
exit	ExitProcess	Terminate execution
open	CreateFile	Create a file or open an existing file
close	CloseHandle	Close a file
read	ReadFile	Read data from a file
write	WriteFile	Write data to a file
lseek	SetFilePointer	Move the file pointer
stat	GetFileAttributesEx	Get various file attributes
mkdir	CreateDirectory	Create a new directory
rmdir	RemoveDirectory	Remove an empty directory
link	(none)	Win32 does not support links
unlink	DeleteFile	Destroy an existing file
mount	(none)	Win32 does not support mount
umount	(none)	Win32 does not support mount
chdir	SetCurrentDirectory	Change the current working directory
chmod	(none)	Win32 does not support security (although NT does)
kill	(none)	Win32 does not support signals
time	GetLocalTime	Get the current time

Tipos de Kernel





Monolítico vs Microkernel