SISTEMAS OPERATIVOS

Las computadoras están equipadas con una capa de software llamada *sistema operativo.* Proporciona a los programas de usuario un modelo de computadora más simple. Por encima del hardware se encuentra el software. La mayoría de las computadoras tienen dos modos de operación: modo kernel y modo usuario.

El sistema operativo se ejecuta en modo kernel. En este modo el sistema operativo tiene acceso a todo el hardware y puede ejecutar cualquier instrucción que la máquina sea capaz de ejecutar. El resto del software se ejecuta en modo usuario en el cual un subconjunto de las instrucciones de maquina está permitido.

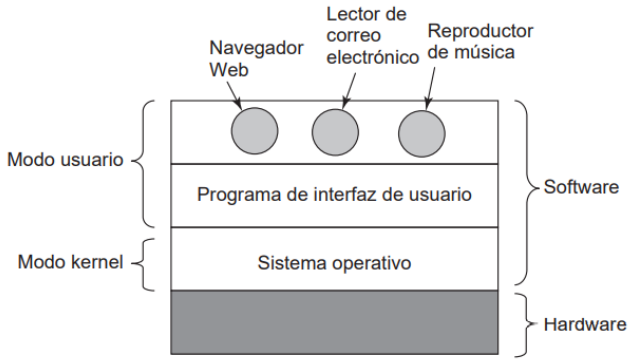


Figura 1-1. Ubicación del sistema operativo.

El programa de interfaz de usuario, Shell, es el nivel más bajo del software en modo usuario y permite la ejecución de otros programas, como un navegador Web, lector de correo electrónico o reproductor de música. Al operar en modo kernel, la CPU puede ejecutar cualquier instrucción de su conjunto de instrucciones y utilizar todas las características del hardware. El sistema operativo opera en modo kernel, lo cual le da acceso al hardware completo.

En contraste, los programas de usuario operan en modo de usuario, el cual les permite ejecutar sólo un subconjunto de las instrucciones y les da acceso sólo a un subconjunto de las características.

¿Qué es un Sistema Operativo?

Es un software que se ejecuta en modo kernel, pero esto no siempre es cierto, ya que cumple dos funciones básicas: proporcionan a los programadores de aplicaciones un conjunto abstracto de recursos simples; y administrar estos recursos de hardware.

El sistema operativo como una maquina extendida

El trabajo del sistema operativo es crear buenas abstracciones para luego implementar y administrar los objetos creados. Una tarea principal del S.O. es ocultar el hardware y presentar a los programadores abstracciones agradables y simples.

El sistema operativo como administrador de recursos

La administración de recursos incluye el multiplexado de recursos en dos formas distintas, en el espacio y en el tiempo:

- En el tiempo los distintos programas o usuarios toman turnos para utilizar un recurso, uno de ellos obtiene acceso al recurso, después otro y así sucesivamente. Por ejemplo, asigna la CPU a distintos programas en distintos tiempos. Determinar quien sigue y durante cuánto tiempo es función del S.O.
- En el espacio es que en vez de que los clientes tomen turnos, cada uno obtiene una parte del recurso, por ejemplo, la memoria principal se divide entre varios programas en ejecución para que cada uno pueda estar residente al mismo tiempo. Esto genera problemas de equidad y protección que le corresponde al S.O resolverlos.

Revisión del hardware de la computadora

Conceptualmente, una computadora personal simple se puede abstraer mediante un modelo de la CPU, la memoria y los dispositivos de E/S están conectados mediante un bus del sistema y se comunican entre sí a través de este bus.

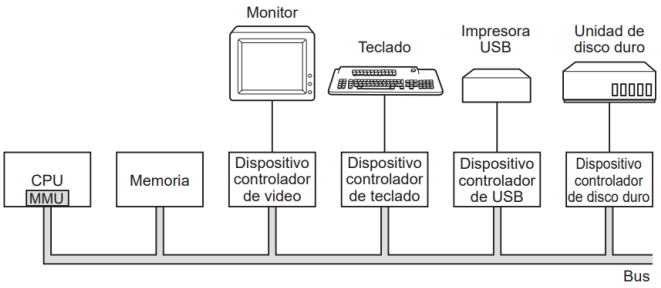


Figura 1-6. Algunos de los componentes de una computadora personal simple.

Procesadores

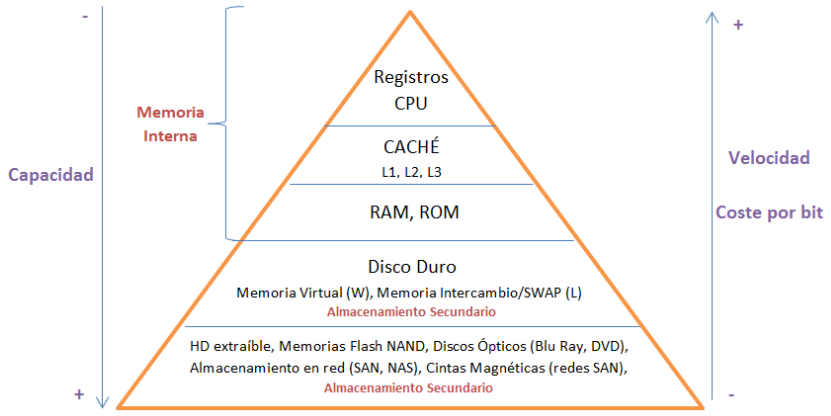
El "cerebro" de la computadora es la CPU, que obtiene las instrucciones de la memoria y las ejecuta. El ciclo básico de toda CPU es obtener la primera instrucción de memoria, decodificarla para determinar su tipo y operandos (Pipeling), ejecutarla y después obtener, decodificar y ejecutar las instrucciones subsiguientes. De esta forma se ejecutan los programas.

Las computadoras tienen varios registros especiales que están visibles para el programador, estos son:

- ***Contador de Programa (program counter):*** el cual contiene la dirección de memoria de la siguiente instrucción a obtener. Una vez que se obtiene esa instrucción, el contador de programa se actualiza para apuntar a la siguiente.
- ***Apuntador de pila (Stack Pointer):*** el cual apunta a la parte superior de la pila (stack) actual en la memoria. La pila contiene un conjunto de valores por cada procedimiento al que se ha entrado, pero del que todavía no se ha salido.
- ***Palabra de estado del programa PSW (Program Status Word):*** Este registro contiene los bits de código de condición, que se asignan cada vez que se ejecutan las instrucciones de comparación, la prioridad de la CPU, el modo (usuario o kernel) y varios otros bits de control.

Memoria

El segundo componente importante en cualquier computadora es la memoria. Esta debe ser rápida y de gran tamaño. El sistema de memoria está construido como una jerarquía de capas.



Dispositivos E/S

Se conoce como dispositivos de entrada/salida a aquellos **aparatos electrónicos que permiten tanto introducir como extraer información** de un sistema. Por ejemplo: un dispositivo de almacenamiento o una impresora multifunción.

Buses

El sistema tiene ocho buses (caché, local, memoria, PCI, SCSI, USB, IDE e ISA), cada uno con una velocidad de transferencia y función distintas. El sistema operativo debe estar al tanto de todos estos buses para su configuración y administración.

LLAMADAS AL SISTEMA

Si yo hago una aplicación y quiero vincularlo con el hardware, lo que hago es una ***llamada a sistema*** (se encuentra en el espacio kernel) para que este haga esa vinculación con las componentes del Sistema Operativo (drivers, manejadores, controladores, etc. del hardware).

PROCEDIMIENTO

Para hacer más entendible el mecanismo de llamadas al sistema, vamos a dar un vistazo rápido a la llamada al sistema ***read***. Como dijimos antes, tiene tres parámetros: el primero especifica el archivo, el segundo apunta al búfer y el tercero proporciona el número de bytes a leer.

read(fd, buffer, nbytes)

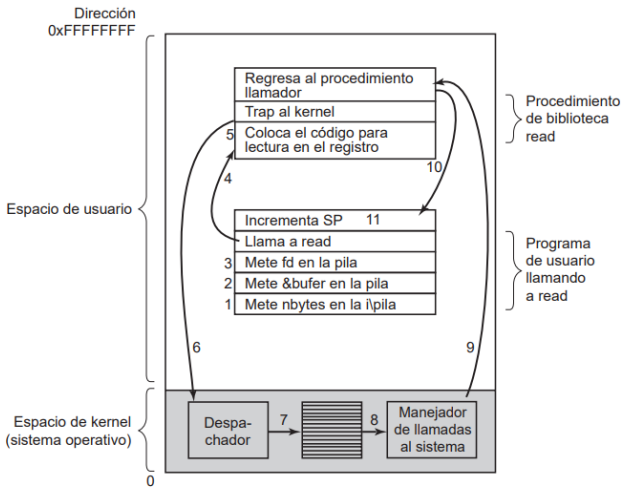


Figura 1-17. Los 11 pasos para realizar la llamada al sistema `read(fd, bufer, nbytes)`.

Lo primero que haces ponemos en la pila los tres argumentos que necesitamos (paso 1 a 3). Después hacemos la llamada a la función *read*, y hacemos la llamada al procedimiento de biblioteca (paso 4). Una vez hecho esto coloca el número de la llamada al sistema, como en un registro (paso 5). Después ejecuta una instrucción TRAP para cambiar de modo usuario a modo kernel y empezar la ejecución en una dirección fija dentro del núcleo (paso 6). Después pasa al manejador correspondiente de llamadas a sistema (paso 7). En ese momento se ejecuta el manejador de llamadas a sistema (paso 8). Una vez que el manejador ha terminado su trabajo, el control se puede regresar al procedimiento de biblioteca que está en espacio de usuario (paso 9). Luego este procedimiento regresa al programa de usuario en la forma usual en que regresan las llamadas a procedimientos (paso 10). Y por ultimo el programa de usuario tiene que limpiar la pila, como lo hace después de cualquier llamada a un procedimiento (paso 11).

ESTRUCTURA DE UN SISTEMA OPERATIVO

Hay seis estructuras que se han probado, estas son: sistemas monolíticos, sistemas de capas, microkernels, sistemas cliente-servidor, máquinas virtuales y exokernels.

SISTEMAS MONOLITICOS

en este diseño, que hasta ahora se considera como la organización más común, todo el sistema operativo se ejecuta como un solo programa en modo kernel. Son procedimientos enlazados entre sí en los cuales cada uno tiene la libertad de llamar a cualquier otro. Si un programa no sabe cómo invocar a una rutina, por ejemplo, podría explotar entonces si da un error todo se me va a la basura famosa pantalla azul.

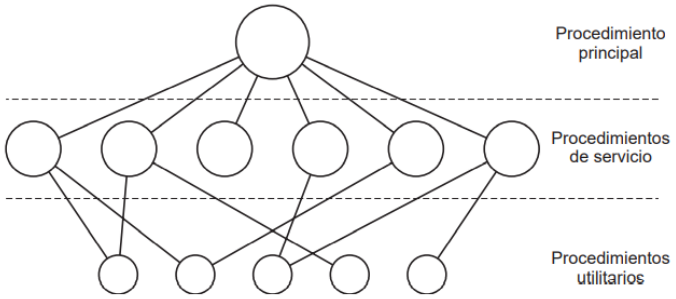


Figura 1-24. Un modelo de estructuración simple para un sistema monolítico.

Estructura:

- Programa principal -> invoca procedimientos de servicios
- Procedimiento de servicios -> invoca a llamas de sistemas
- Procedimiento utilitarios -> ayudan a los procedimientos de servicios

SISTEMAS DE CAPAS

Consiste en ordenar el sistema operativo en una jerarquía de capas, consta de 6 capas.

5 – El operador (Se localiza el proceso que opera)
4 – Programas de usuario (No importa la gestión del kernel)
3 – Administración de la entrada/salida (Guarda en buffer el flujo de información entre ellos)
2 – Comunicación operador-proceso (comunicación con el usuario)
1 – Administración de memoria (Administración de memoria para los procesos)
0 – Asignación del procesador y multiprogramación

Cada capa se comunica con la adyacente. Es más lento porque cada tarea tiene que usar un servicio de la capa anterior hasta llegar a la capa de asignación de proceso y multiprogramación.

ESTRUCTURA DE MICROKERNEL

El objetivo es hacer el microkernel lo más liviano posible y cada vez que necesite gestionar memoria lo pido como un servicio a un proceso en modo usuario. La idea es sacar las funcionalidades del kernel y llevarlas a modo usuario. El microkernel solo se encarga de la planificación de hilos y no de procesos esto posibilita multitarea. Siendo así el microkernel el que planifica todo el código que corre el sistema.

ESTRUCTURA DE MAQUINAS VIRTUALES

Básicamente se multiplexa el hardware. Serian como dos capas:

- Capa de multiprogramación -> Hardware en donde se ejecuta el kernel
- Capa de maquina extendida -> Abstracción del hardware

Entonces yo tengo un proceso que usa la parte bonita del sistema operativo y cree que está solo, es decir aparentando que cada terminal (proceso) posee su propia maquina real la cual es proporcionada por la capa de hardware que se encarga de multiprogramar muchas máquinas virtuales sobre una maquina física.