

CAPÍTULO 1 PANORAMA GENERAL.

¿Qué es del hardware sin el software?

¿Qué se puede decir del software sin hardware?

El software de computadora puede dividirse a grandes rasgos en dos tipos:

1. Programas de sistema.
2. Programas de aplicación.

El programa de sistema más importante es el Sistema Operativo (SO).

El SO controla todos los recursos de la computadora y establece la base sobre la que pueden escribirse las aplicaciones.

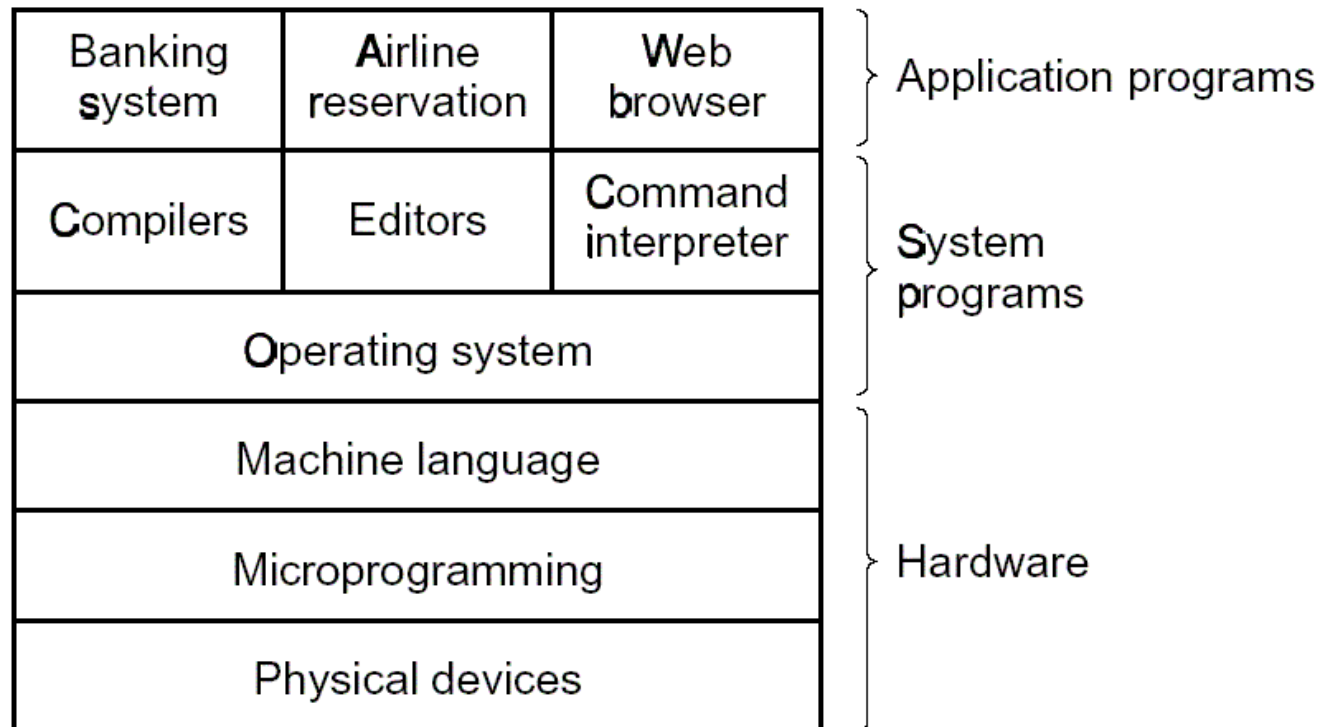


Figure 1-1. A computer system consists of hardware, system programs, and application programs.

Una función importante del SO es ocultar toda la complejidad inherente al uso y manejo de los dispositivos del sistema de cómputo y ofrecer al programador un conjunto de instrucciones más cómodo con el que pueda trabajar.

Ejemplo de lo anterior son el conjunto de **llamadas al sistema**.

Por otro lado, el SO es la porción de SW que se ejecuta en **modo kernel** o **modo supervisor**, y está protegido por el HW contra la intervención del usuario.

1.1 Vistas de un Sistema Operativo.

La mayoría de los usuarios de computadora han tenido experiencia con algún SO, pero no es fácil precisar con exactitud qué es un SO.

De manera general, existen dos vistas o funciones de un SO:

1. El SO como máquina extendida.
2. El SO como administrador de recursos.

1.1.1 El SO como máquina extendida.

La arquitectura¹ de la mayor parte de las computadoras en el nivel del lenguaje de máquina es primitiva y difícil de programar.

¹ Conjunto de instrucciones, organización de la memoria y la E/S y la estructura de los buses.

Considérense los aspectos que involucra la lectura de datos de una unidad de disco flexible.

El programa que oculta la verdad acerca del HW y presenta al programador una vista sencilla y funcional de la máquina es el SO.

El SO presenta una interfaz sencilla orientada a archivos, oculta muchos de los asuntos referentes a las interrupciones, temporizadores, administración de memoria y otras funciones de bajo nivel.

La abstracción que el SO ofrece en cada caso es más sencilla y fácil de usar que el HW subyacente.

En este tipo de vista, la función del SO es presentar al usuario el equivalente de una **máquina extendida** o **máquina virtual** más fácil de programar que el HW subyacente.

Este tipo de visión se conoce como visión descendente.

1.1.2 El SO como administrador de recursos.

Esta es una visión ascendente y postula que el SO está ahí para administrar todos recursos del sistema.

En esta visión, la función de un SO es asegurar un reparto ordenado y controlado de:

1. procesadores

2. memorias
3. dispositivos de E/S

entre los diferentes procesos que compiten por ellos.

¿Qué pasa cuando dos o más procesos intentan usar un mismo recurso de manera simultánea?

1.2 Historia de los Sistemas Operativos.

Material de lectura.

Solamente se considerarán algunos puntos importantes:

- La primera computadora digital verdadera fue diseñada por el matemático inglés Charles Babbage.

- La historia de los SO está íntimamente ligada a la de las generaciones de las computadoras.
- Fred Brooks, uno de los diseñadores del OS/360².
- Con el OS/360 surge el concepto de **multiprogramación**.

² Primera máquina en usar a pequeña escala circuitos integrados.

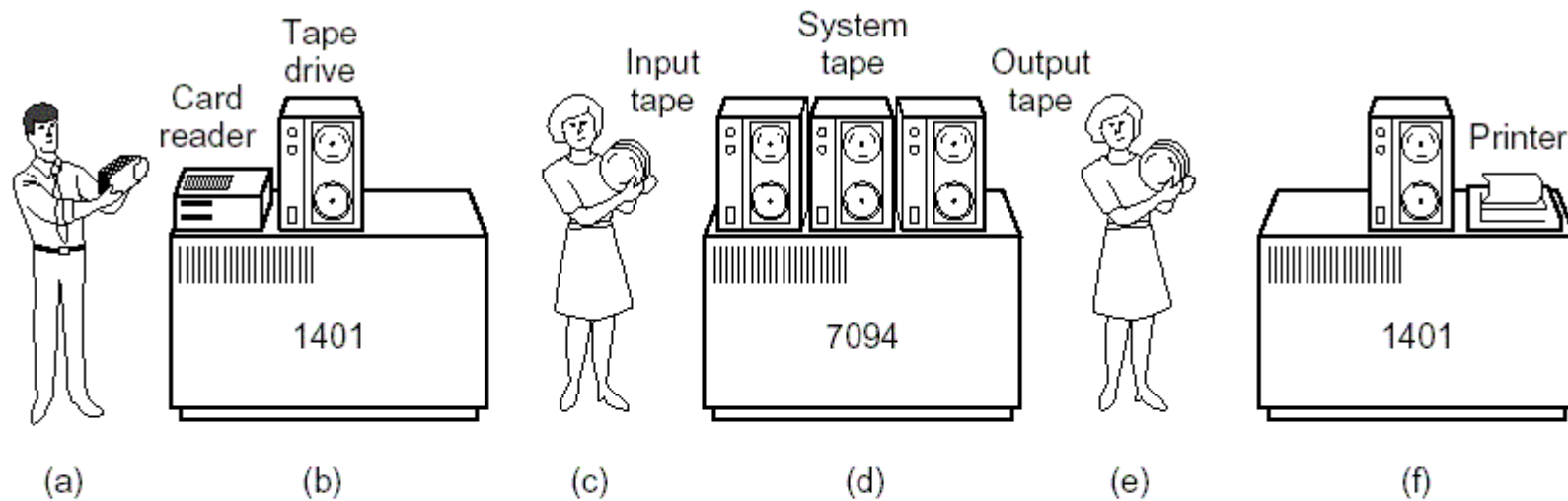


Figure 1-2. An early batch system. (a) Programmers bring cards to 1401. (b) 1401 reads batch of jobs onto tape. (c) Operator carries input tape to 7094. (d) 7094 does computing. (e) Operator carries output tape to 1401. (f) 1401 prints output.

- Concepto de **tiempo compartido** (3^a generación).

1.3 Conceptos de Sistemas Operativos.

La interfaz entre el SO y los programas de usuario está definida por el conjunto de operaciones extendidas que el SO ofrece. Estas instrucciones se denominan **llamadas al sistema**.

1.3.1 Procesos.

La instancia en ejecución de un programa se denomina **proceso**.

Cada proceso tiene asociado un espacio de direcciones.

Un **espacio de direcciones** es una lista de posiciones de memoria con un mínimo y un máximo que el proceso utiliza.

Cada proceso tiene asociado tres segmentos:

1. Texto (programa ejecutable).
2. Datos.
3. Pila.

A cada proceso se le asocia también un conjunto de registros, entre ellos están el contador de programa y el apuntador de la pila.

Toda la información de un proceso, aparte del contenido de su propio espacio de direcciones se almacena en una tabla del SO llamada tabla de procesos.

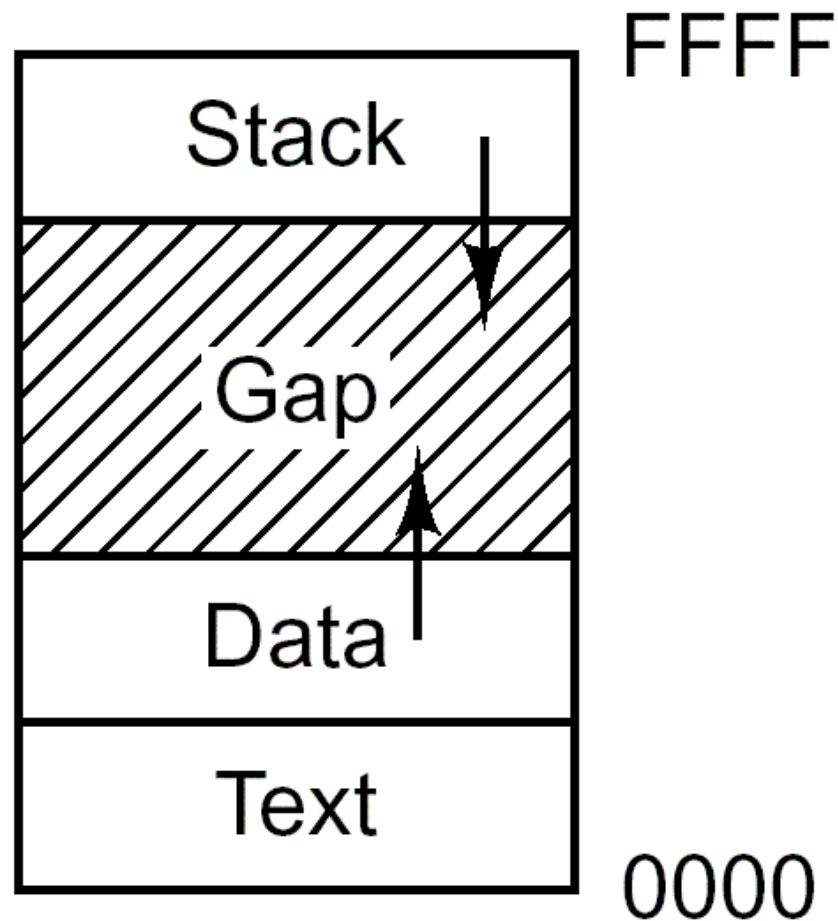


Figure 1-11. Processes have three segments: text, data, and stack. In this example, all three are in one address space, but separate instruction and data space is also supported.

La **tabla de procesos** es un arreglo o lista enlazada de estructuras, una por cada proceso existente en ese momento.

Las llamadas claves al sistema de administración para procesos, son las que se ocupan de la creación y terminación de procesos.

Considérese el proceso denominado shell (intérprete de comandos) y la tarea de compilación de un programa.

Un proceso puede crear uno o más procesos distintos (**procesos hijos**) y estos a su vez pueden hacer lo mismo.

Los procesos relacionados que están cooperando para un fin, se comunican a través de mecanismos de IPC.

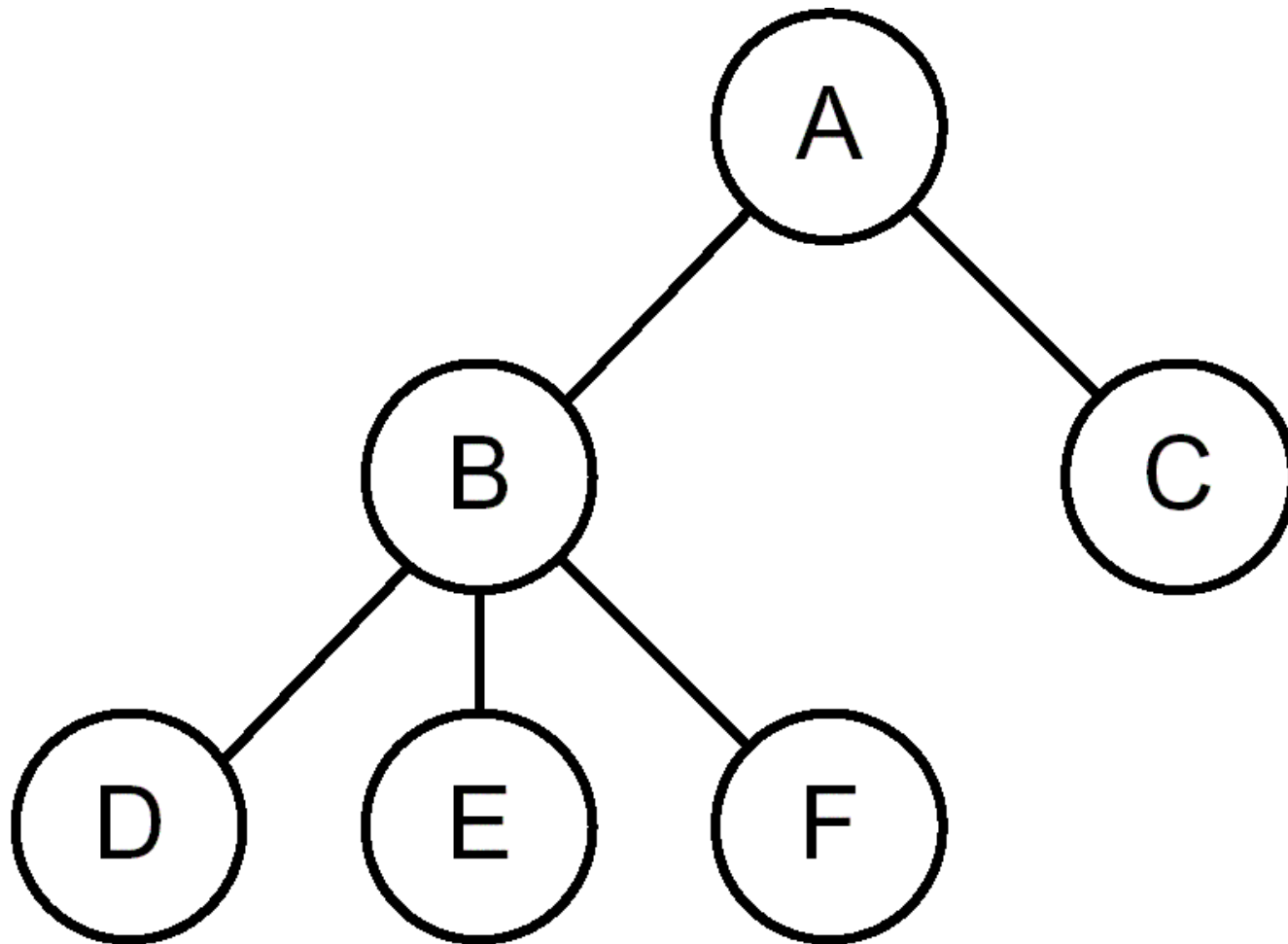


Figure 1-5. A process tree. Process *A* created two child processes, *B* and *C*. Process *B* created three child processes, *D*, *E*, and *F*.

Cada proceso iniciado en un sistema UNIX por ejemplo, tiene asociado el **uid** (identificador de usuario) de quien lo inició.

Un proceso hijo tiene el mismo uid de su padre.

Existe un uid especial llamado **superusuario**, mismo que tiene facultades y privilegios especiales.

1.3.2 Archivos.

Otra categoría amplia de llamadas al sistema se relaciona con el sistema de archivos.

La principal función de este tipo de llamadas al sistema es presentar al programador un modelo abstracto y útil de archivos.

UNIX tiene el concepto de directorio como un mecanismo para guardar y agrupar los archivos.

Una estructura de directorio es un modelo organizacional de información (archivos).

Este modelo da pie a una jerarquía de archivos y directorios denominada **sistema de archivos**.

Las jerarquías de procesos y de archivos están organizadas como árboles.

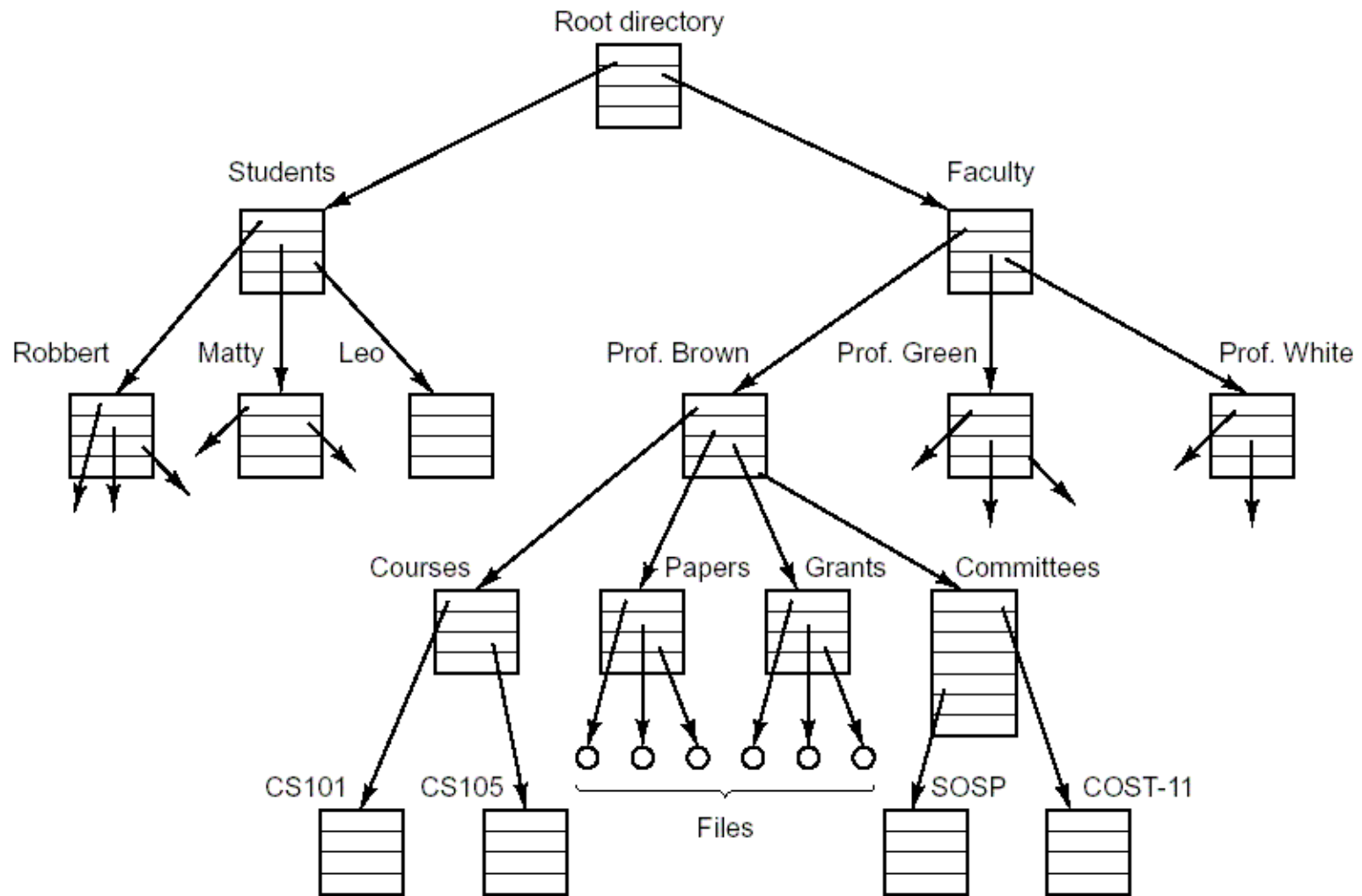


Figure 1-6. A file system for a university department.

En el sistema de archivos son importantes los conceptos de ruta, directorio raíz, directorio de trabajo, códigos de protección (*rwX*), descriptor de archivo (entero), etc.

Un concepto importante también en UNIX es el de **sistema de archivos montado**.

Considérese el caso de una unidad de disco flexible.

Para evitar la dependencia de dispositivos, UNIX no permite anteponer a los nombres de ruta un nombre o número de unidad.

Windows y MS-DOS son ejemplos de este tipo de dependencias de dispositivos.

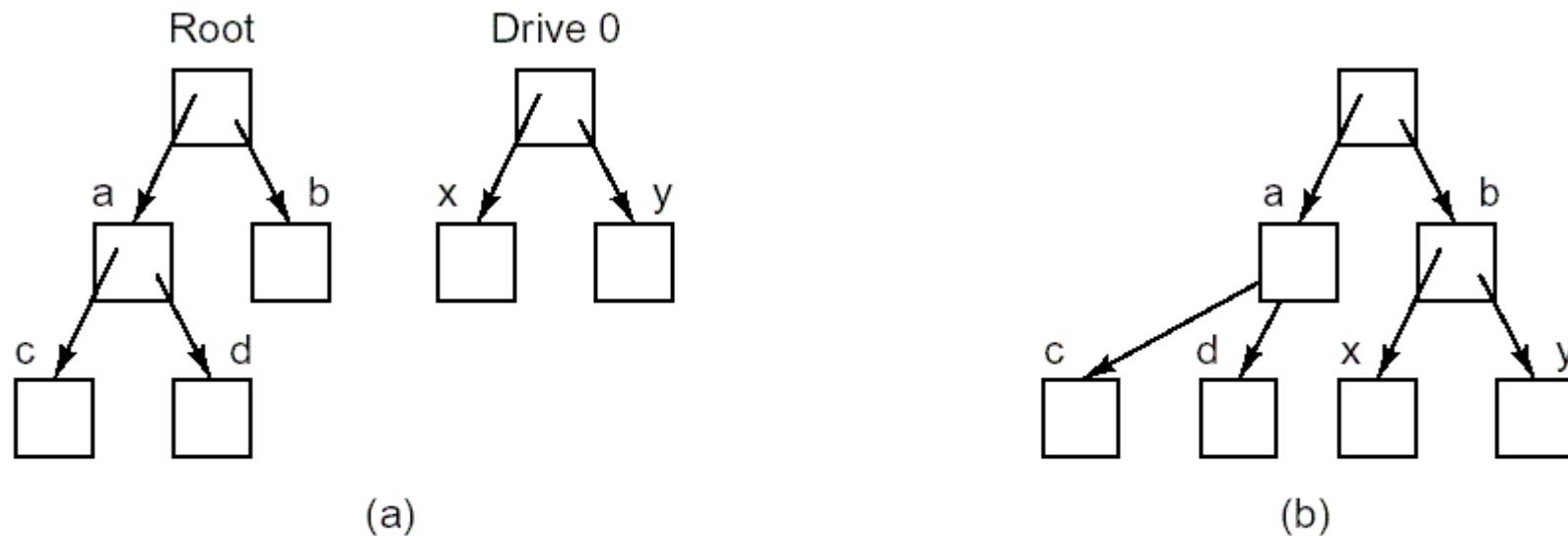


Figure 1-7. (a) Before mounting, the files on drive 0 are not accessible. (b) After mounting, they are part of the file hierarchy.

Los archivos especiales sirven para hacer que los dispositivos de E/S semejen archivos.

Existen también dos tipos de archivos especiales:

1. Archivos especiales por bloques: modelar dispositivos que consisten en una colección de bloques direccionables.
2. Archivos especiales por caracteres: usados para modelar impresoras, módems y otros dispositivos orientados al flujo de caracteres.

Una **tubería** (*pipe*) es una especie de pseudo archivo que puede servir para conectar dos procesos.

La tubería es un mecanismo útil, aunque no el mejor, de comunicación entre procesos (IPC).

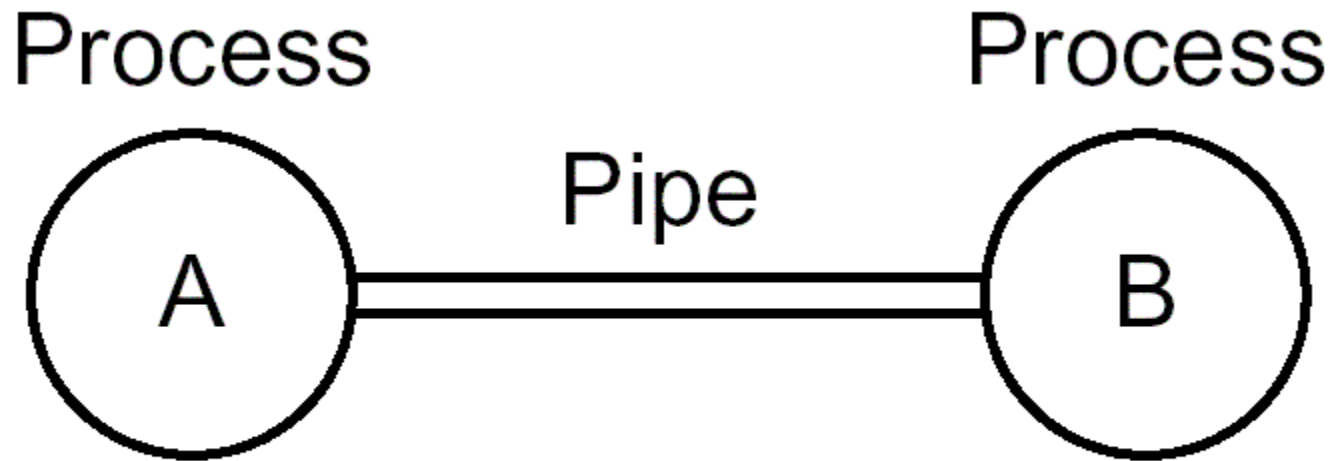


Figure 1-8. Two processes connected by a pipe.

1.3.3 El shell.

El shell es la interfaz primaria entre un usuario sentado ante su terminal y el sistema operativo.

Estrictamente hablando, un shell NO es parte del SO (que es el código que ejecuta las llamadas al sistema), pero utiliza muchas de sus características.

1.4 Llamadas al sistema.

Ejemplos de llamadas al sistema (material de lectura: sección 1.4 - Tanenbaum y capítulo 7 - Kernighan).

1.5 Estructuras de los Sistemas Operativos.

Se examinarán cuatro estructuras de los sistemas operativos, no son las únicas pero sí las más comunes:

1. Sistemas monolíticos.

2. Sistemas por capas.
3. Máquinas virtuales.
4. Sistemas cliente – servidor.

1.5.1 Sistemas monolíticos.

Es quizá la organización más común.

La estructura consiste en que no hay estructura, por eso tiende a llamarse “El gran desorden”.

El SO se escribe como una colección de procedimientos, en donde cada uno de los cuales puede invocar a cualquiera de los otros cuando necesite hacerlo.

Las llamadas al sistema se solicitan colocando los parámetros en lugares bien definidos, como en registros específicos o en la pila y ejecutando después una instrucción de trampa conocida como llamada al kernel o llamada al supervisor.

Una **llamada al kernel** conmuta a la máquina del modo usuario al modo kernel y transfiere el control al SO.

La estructura básica del SO es:

1. Un programa principal que invoca el servicio.
2. Un conjunto de procedimientos de servicio.
3. Un conjunto de procedimientos de utilería.

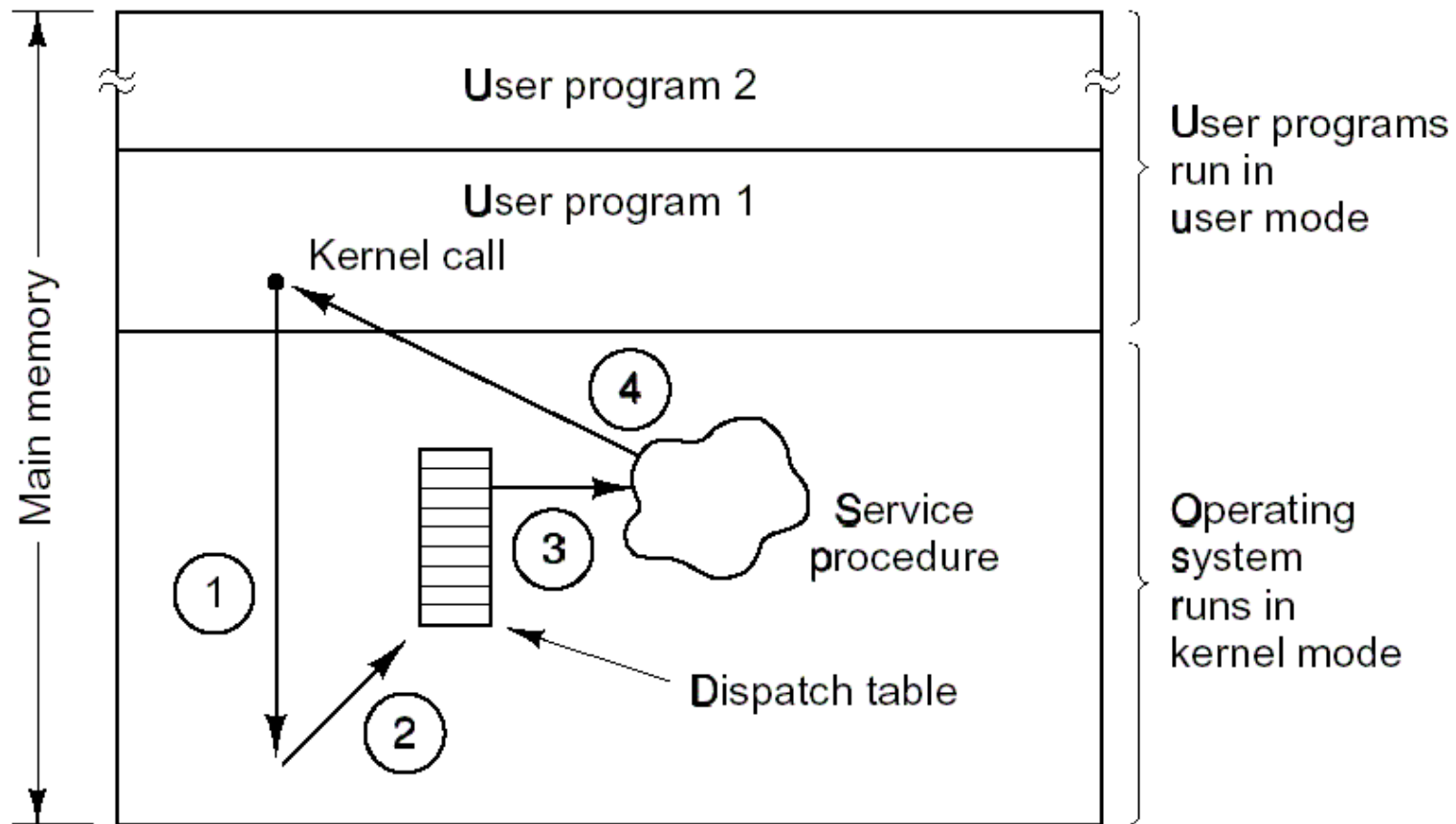


Figure 1-16. How a system call can be made: (1) User program traps to the kernel. (2) Operating system determines service number required. (3) Operating system calls service procedure. (4) Control is returned to user program.

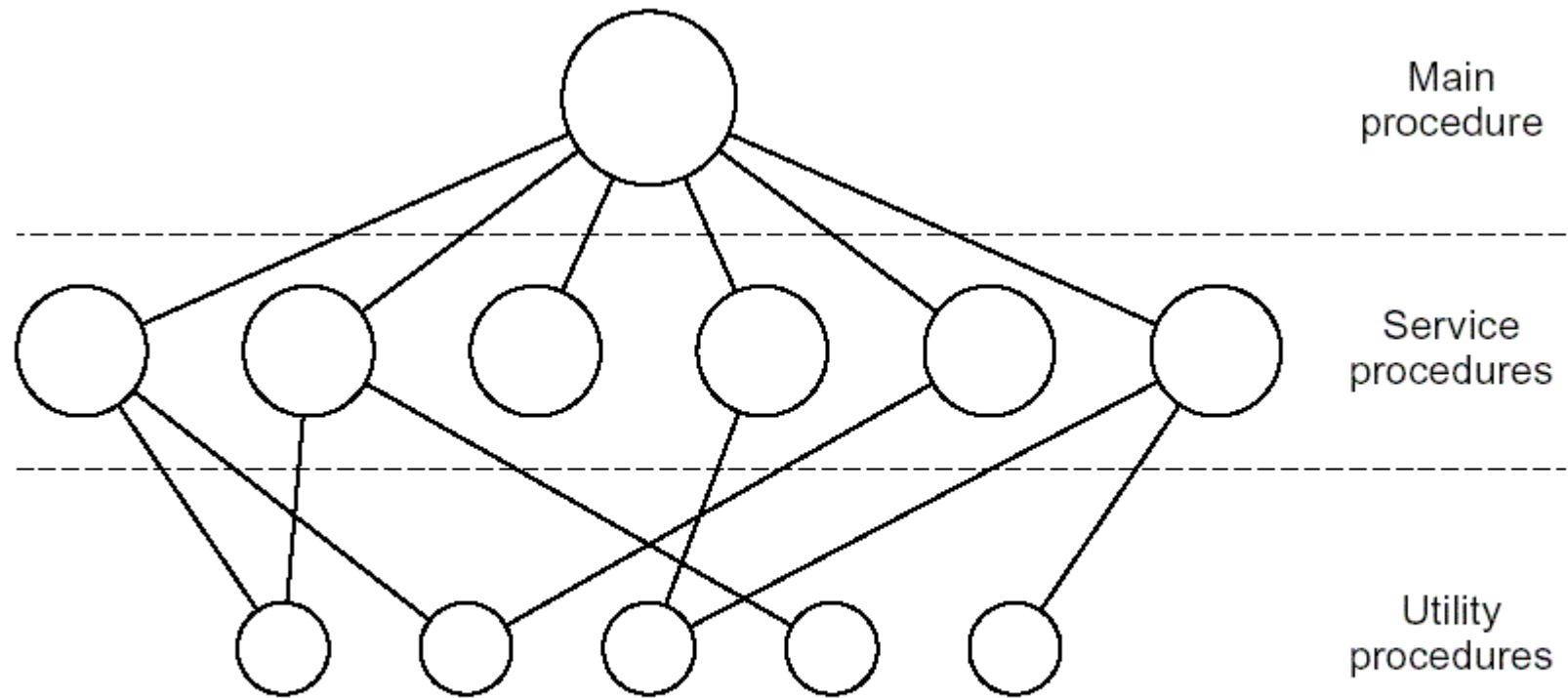


Figure 1-17. A simple structuring model for a monolithic system.

1.5.2 Sistemas por capas.

Consiste en una generalización del enfoque anterior al organizar al SO como una jerarquía de capas.

El primer sistema que tuvo esta estructura fue el sistema THE (Technische Hogeschool Eindhoven) en los países bajos (Holanda) por Dijkstra y sus estudiantes.

Una forma más generalizada del concepto de capas estuvo presente en el sistema MULTICS.

MULTICS estaba organizado en anillos concéntricos en vez de en capas, siendo los interiores más privilegiados que los exteriores.

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

Figure 1-18. Structure of the THE operating system.

1.5.3 Máquinas virtuales.

La esencia de este tipo de sistemas radica en el **monitor de máquina virtual**, éste se ejecuta directamente sobre el HW y

realiza la multiprogramación proporcionando una o más máquinas virtuales a la capa superior.

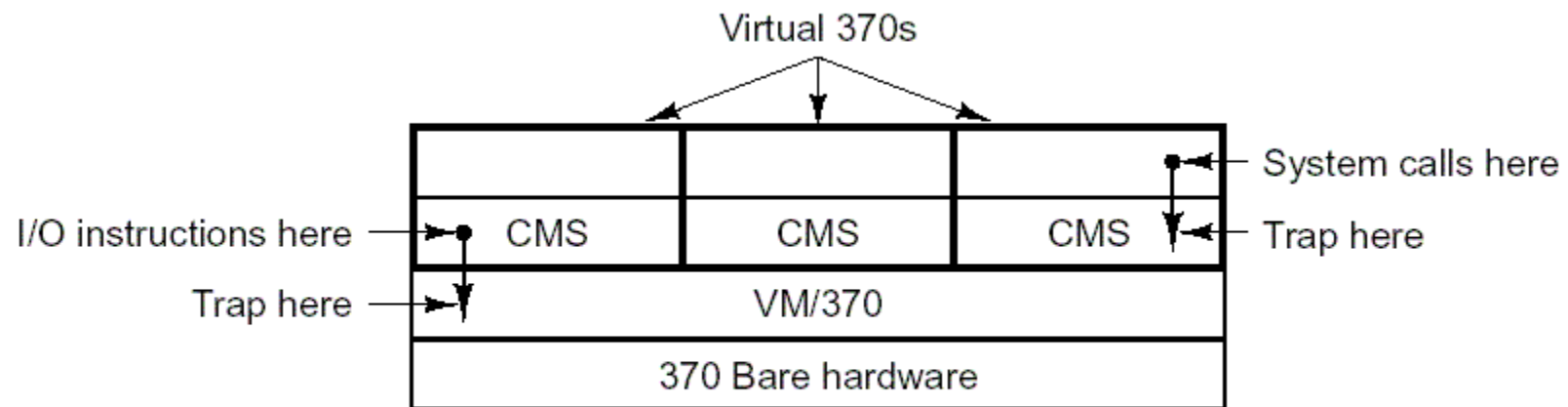


Figure 1-19. The structure of VM/370 with CMS.

Es importante mencionar que las máquinas virtuales no son máquinas extendidas, con abstracciones como servicios y archivos,

en realidad son copias exactas del HW, incluido el modo de kernel, el modo de usuario, la E/S, las interrupciones, etc.

CMS viene de Sistema de Monitoreo de Conversaciones.

Un sistema de este tipo separa por completo las funciones de multiprogramación y de suministro de una máquina extendida.

¿Qué es el modo 8086 virtual de Pentium?

1.5.4 Modelo cliente – servidor.

Una tendencia que se tiene desde hace tiempo respecto a los SO modernos, es hacer cada vez más pequeño el código del SO y trasladar otras partes del código a capas superiores.

Este concepto recibe el nombre de **micro kernel** y consiste en implementar la mayor parte de las funciones del SO en procesos de usuario.

Para solicitar un servicio, un **proceso de usuario** o **proceso cliente** envía la solicitud a un **proceso servidor**, el cual realiza el trabajo y devuelve la respuesta.

Al dividir el SO en partes, cada una de las cuales sólo se encarga de una faceta del sistema, cada parte es pequeña y manejable, y si una se cae, las otras siguen funcionando debido a que funcionan como procesos de usuario y no en modo kernel.

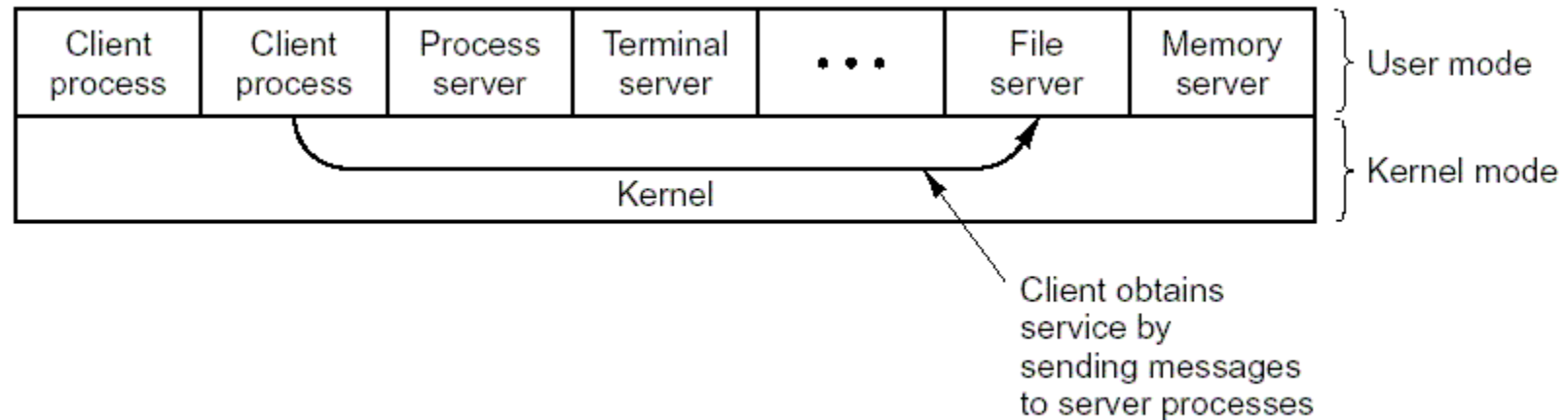


Figure 1-20. The client-server model.

Otra ventaja adicional de este esquema, es su adaptabilidad para usarse en sistemas distribuidos.

Si un cliente se comunica con un servidor enviándole mensajes, el cliente no necesita saber si el mensaje será atendido localmente o si se envía a través de la red.

El cliente envía una solicitud y obtiene una respuesta, no importa si es desde la misma máquina o de otra en la red.

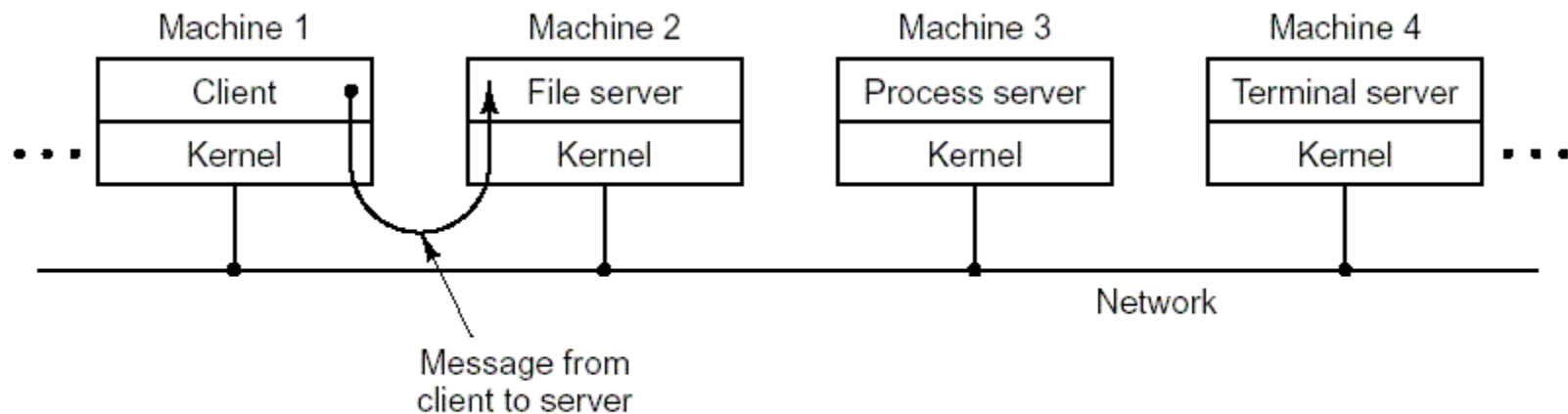


Figure 1-21. The client-server model in a distributed system.

Este modelo, en el que el kernel sólo se encarga del transporte de mensajes no es tan realista y mucho menos tan sencillo como parece.

Finalmente es importante mencionar que los SO por lo regular tienen 4 componentes principales:

1. Administración de procesos.
2. Administración de memoria.
3. Administración de E/S.
4. Administración de archivos.

Por lo que el objetivo general del curso será cubrir los elementos principales que se siguen en estos esquemas de administración.