Sistemas Operativos

Estructuras de sistemas

Armando Arce, Tecnológico de Costa Rica, arce@itcr.ac.cr

Tecnológico de Costa Rica

Estructuras del sistema operativo

Un sistema operativo puede ser visto desde varios puntos de vista.

- Uno de ellos se centra en los servicios que el sistema proporciona;
- otro, en la interfaz disponible para los usuarios y programadores;
- un tercero, en sus componentes y sus interconexiones.

Servicios del sistema operativo

El sistema operativo presta ciertos servicios a los programas y a los usuarios de dichos programas. Estos servicios se proporcionan para comodidad del programador, con el fin de facilitar la tarea de desarrollo:

- Ejecución de programas: Se debe poder cargar un programa en memoria y ejecutar dicho programa.
- Operaciones de E/S: Los usuarios no pueden controlar de modo directo los dispositivos de E/S; por tanto, el sistema operativo debe proporcionar medios para realizar la E/S.

Servicios del sistema operativo

- Manipulación de archivos: Se brinda el servicio de leer, escribir, crear y borrar tanto archivos como directorios. También se incluyen permisos de acceso.
- Comunicaciones: Las comunicaciones entre procesos se pueden implementar utilizando memoria compartida o mediante paso de mensajes.
- Detección de errores: Para cada tipo de error, el sistema operativo debe llevar a cabo la acción apropiada para asegurar su funcionamiento correcto. Los errores pueden ser de memoria, alimentación de dispositivo, conexión a la red, o del programa de usuario.

Servicios del sistema operativo

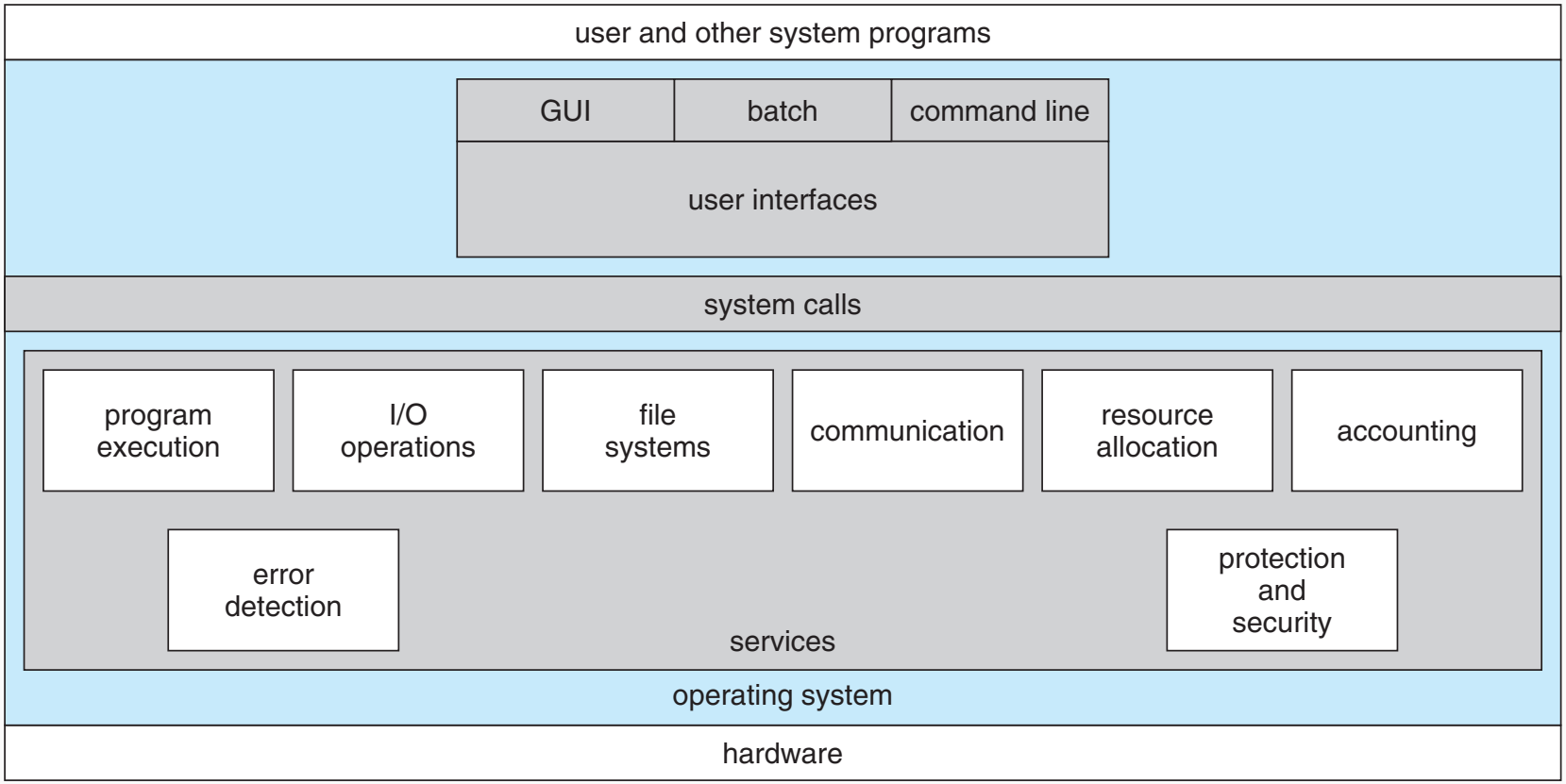

Figure 2.1 A view of operating system services.

Figure 1:

Servicios del sistema operativo

Existe otro conjunto de funciones del sistema operativo que garantizan la eficiencia del propio sistema:

- Asignación de recursos: El sistema operativo administra muchos tipos diferentes de recursos; algunos pueden disponer de código software especial que administre su asignación, mientras otros pueden tener código que administre e forma más general su solicitud y liberación.
- Contabilización: Se lleva un registro de qué usuarios emplean qué clase de recursos. Estas estadísticas se utilizan para reconfigurar el sistema o facturación a usuarios.
- Protección y seguridad: La protección implica asegurar que todos los accesos a los recursos estén controlados. También es importante garantizar la seguridad del sistema frente a posibles intrusos.

Llamadas al sistema

Las llamadas al sistema proporcionan una interfaz con la que invocar los servicios que el sistema operativo ofrece.

- Estas llamadas están disponibles como rutinas escritas en C o C++, aunque algunas pueden estar escritas en ensamblador.

Llamadas al sistema

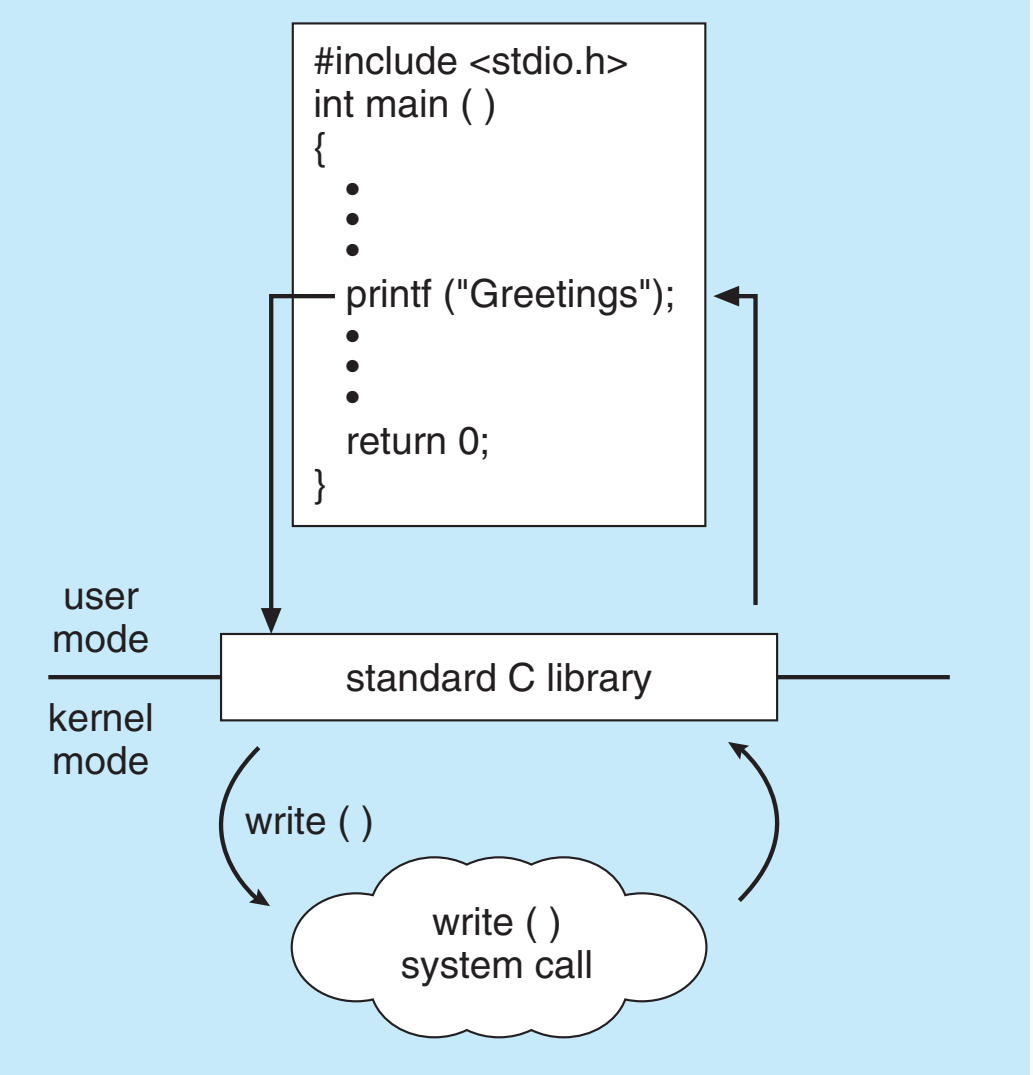

Figure 2:

Llamadas al sistema

Normalmente, los desarrolladores de aplicaciones diseñan sus programas utilizando una API (interfaz de programación de aplicaciones).

- La API especifica un conjunto de funciones que el programador de aplicaciones puede usar, indicándose los parámetros que hay que pasar a cada función y los valores de retorno que el programador debe esperar.

Llamadas al sistema

Las ventajas de usar un API son:

- Portabilidad: Se puede compilar y ejecutar el programa en cualquier sistema que soporte la misma API.
- Mayor grado de facilidad: A veces resulta más difícil trabajar con las propias llamadas al sistema y exige un grado mayor de detalle.

Sistema de soporte en tiempo de ejecución

El sistema de soporte en tiempo de ejecución (un conjunto de funciones biblioteca que suele incluirse con los compiladores) de la mayoría de los lenguajes de programación proporciona una interfaz de llamadas al sistema que sirve como enlace con las llamadas al sistema disponibles en el sistema operativo.

Sistema de soporte en tiempo de ejecución

La interfaz de llamadas al sistema intercepta las llamadas a función dentro de las API e invoca la llamada al sistema necesaria. Habitualmente, cada llamada al sistema tiene asociado un número y la interfaz de llamadas al sistema mantiene una tabla indexada según dichos números.

- La API oculta al programador la mayor parte de los detalles de interfaz del sistema operativo, los cuales son administrados por la biblioteca de soporte en tiempo de ejecución.

Paso de parámetros al sistema operativo

Para pasar parámetros al sistema operativo se emplean tres métodos generales:

- Mediante registros: consiste en pasar los parámetros mediante registros del CPU. Sin embargo, a veces existen más parámetros que registros.

Paso de parámetros al sistema operativo

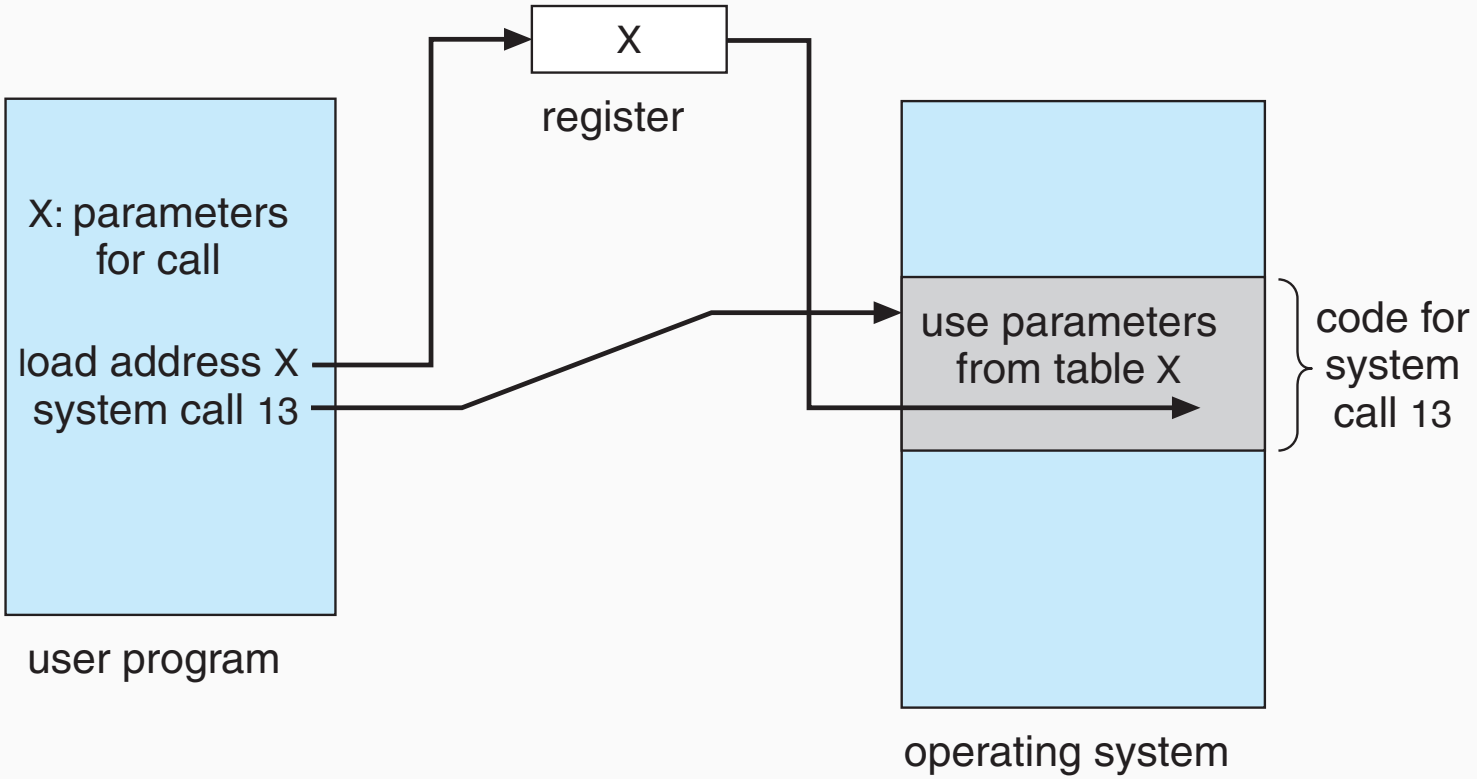

Figure 2.7 Passing of parameters as a table.

Figure 3:

Paso de parámetros al sistema operativo

- Mediante bloque: los parámetros se almacenan en un bloque o tabla, en memoria, y la dirección del bloque se pasa como parámetro en un registro
- Mediante la pila: también se pueden colocar los parámetros en la pila del proceso y el sistema operativo se encargará de extraer de la pila dichos parámetros.

Operación en modo dual

La mayoría de los computadoras tienen dos modos de operación, modo *kernel* y modo *usuario*. A esto se le conoce como *modo dual de operación*.

Operación en modo dual

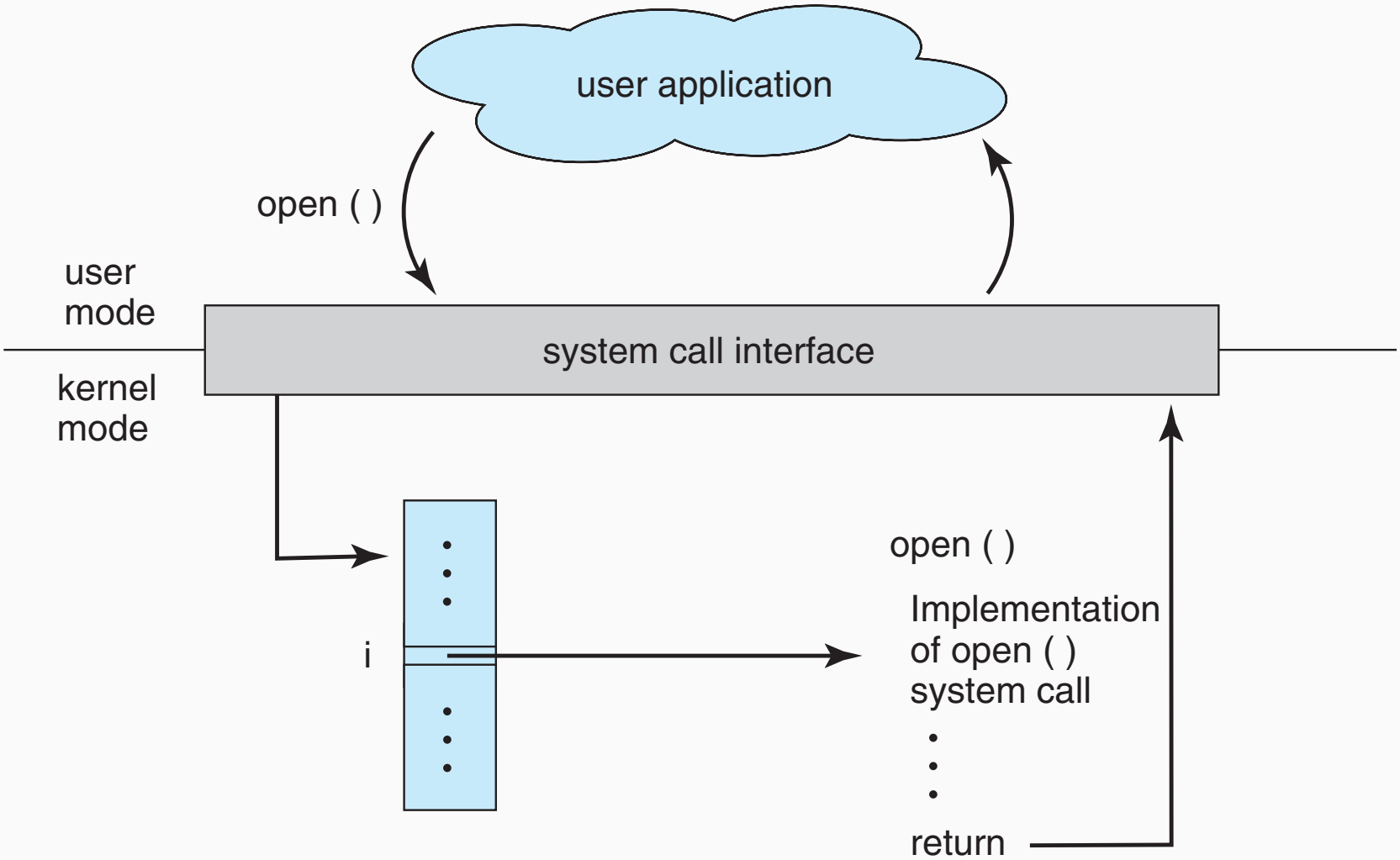

Figure 2.6 The handling of a user application invoking the `open()` system call.

Operación en modo dual

El sistema operativo se ejecutan en modo *kernel* (supervisor ó privilegiado), en el cual tiene acceso completo a todo el hardware del computador.

- El resto del software se ejecuta en modo usuario y solamente pueden ejecutar un subconjunto de instrucciones de la máquina.
- Las instrucciones de E/S no son permitidas en modo usuario.

Operación en modo dual

El modo dual proporciona el medio para realizar la protección del sistema por parte de usuarios que puedan cometer errores, y a su vez para proteger a los usuarios de los errores de otros usuarios.

- Para implantar este mecanismo el hardware cuenta con un bit, llamado *bit de modo*, que indica el modo actual.
- Cuando se ejecuta un programa de usuario y se produce una llamada al sistema, el sistema cambia del modo usuario al modo *kernel* modificando el bit.
- Cuando se termina de procesar la llamada, el sistema regresará el control al programa de usuario cambiando del modo *kernel* al modo usuario.

Estructura del sistema

Un método común de diseñar un sistema operativo consiste en dividir la tarea en componentes mas pequeños, en lugar de tener un sistema monolítico. Existen diferentes formas de estructurar estos componentes, tal como se vera a continuación:

Estructura simple

Muchos sistemas no tienen una estructura bien definida y los niveles de funcionalidad no están separados.

- Esta estructura puede provocar que el sistema sea vulnerable a programas erróneos, lo que hace que el sistema completo falle cuando los programas de usuario fallan.
- Además este tipo de sistemas son difíciles de implementar y mantener.

Estructura en niveles

El sistema operativo se divide en una serie de capas (niveles).

- El nivel inferior es el hardware, el nivel superior es la interfaz de usuario.
- Los niveles se diseñan de modo que cada uno usa funciones y servicios de los niveles inferiores.

Estructura en niveles

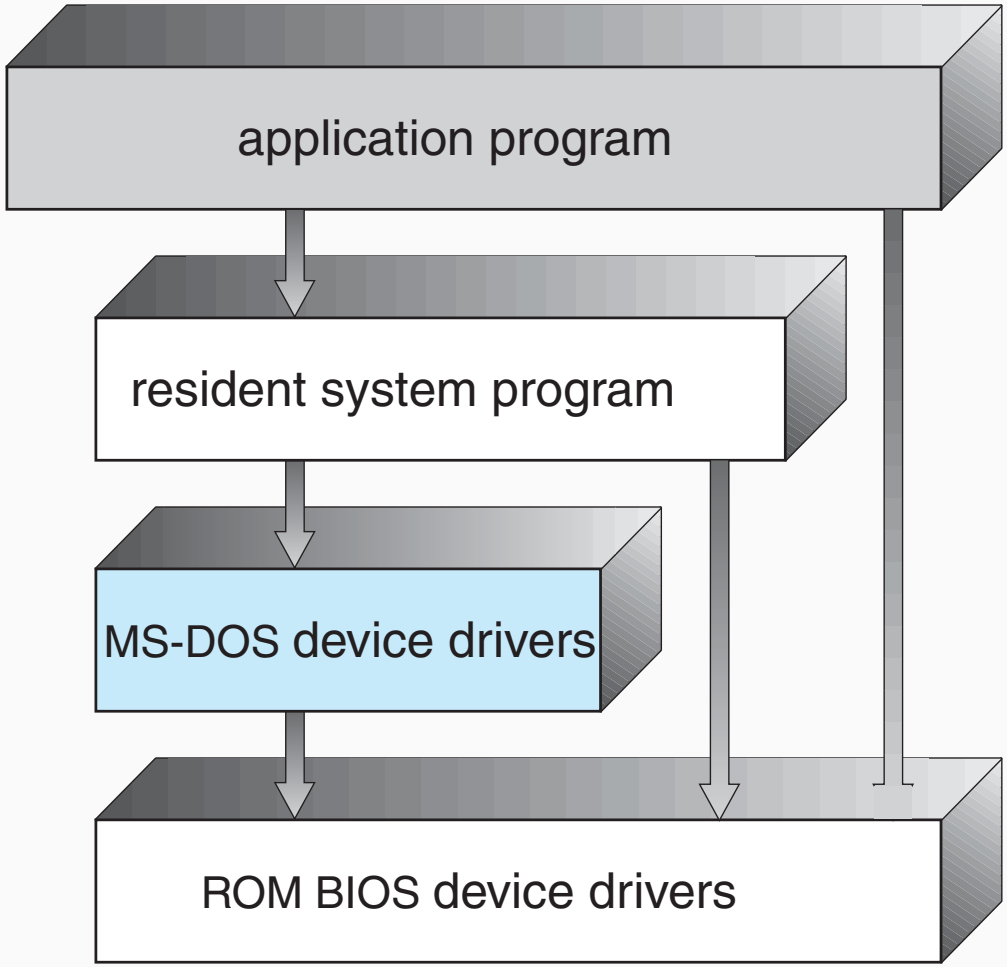

Figure 2.11 MS-DOS layer structure.

Estructura en niveles

La principal ventaja del método de niveles es la simplicidad de construcción y depuración.

- Si se va depurando por niveles, un error tendrá que estar localizado en el nivel que se está depurando, dado que los niveles inferiores a él ya se han depurado.

Estructura en niveles

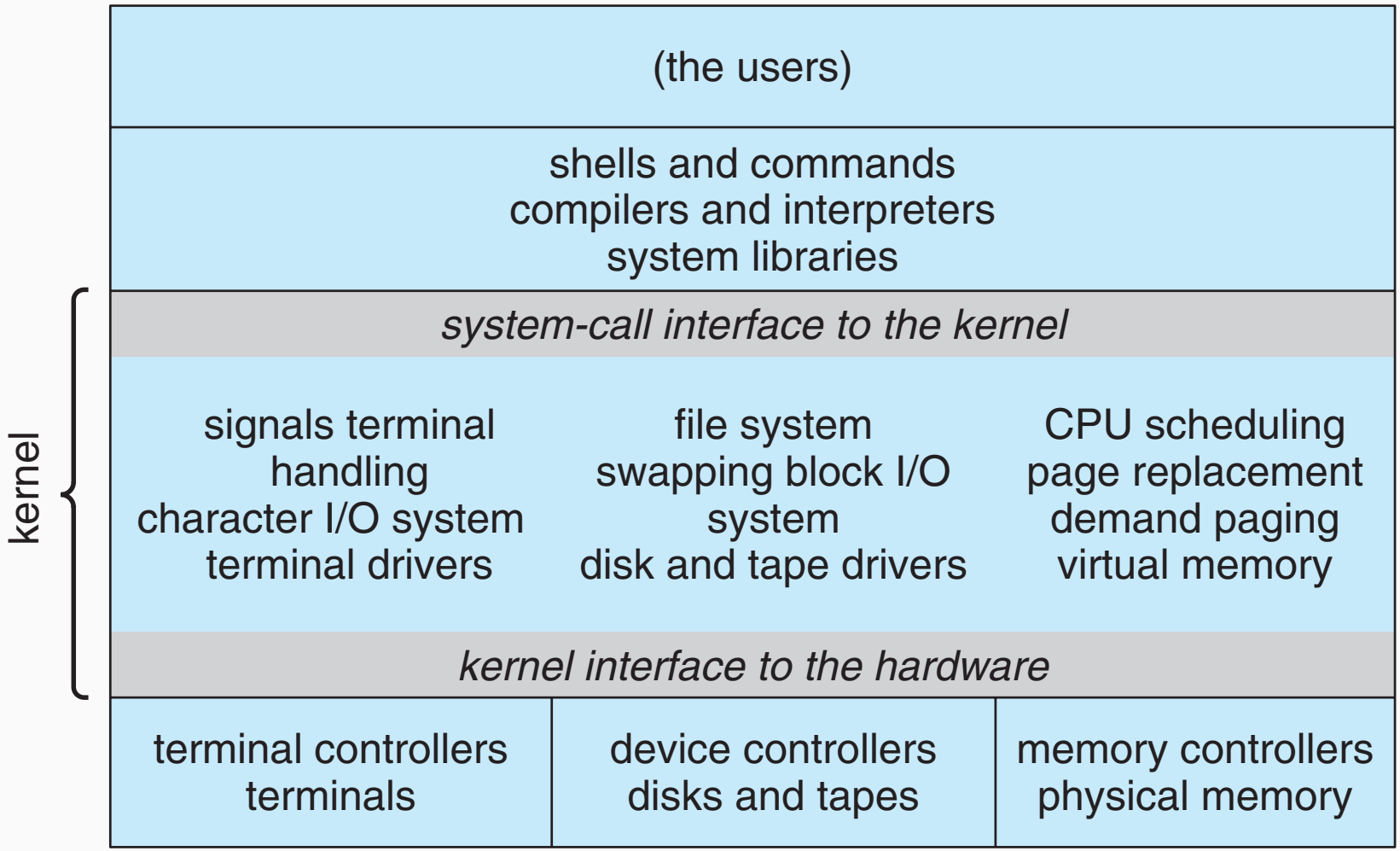

Figure 2.12 Traditional UNIX system structure.

Estructura en niveles

La principal dificultad de este tipo de sistema es la de definir apropiadamente los diferentes niveles, pues es necesario realizar una planificación cuidadosa.

- Otro problema es que la estructura en niveles tiene a ser menos eficiente que otras pues cada nivel añade una carga de trabajo adicional en cada llamada al sistema.
- Por eso los sistemas actuales utilizan menos niveles, con mas funcionalidad por cada nivel.

Estructura en niveles

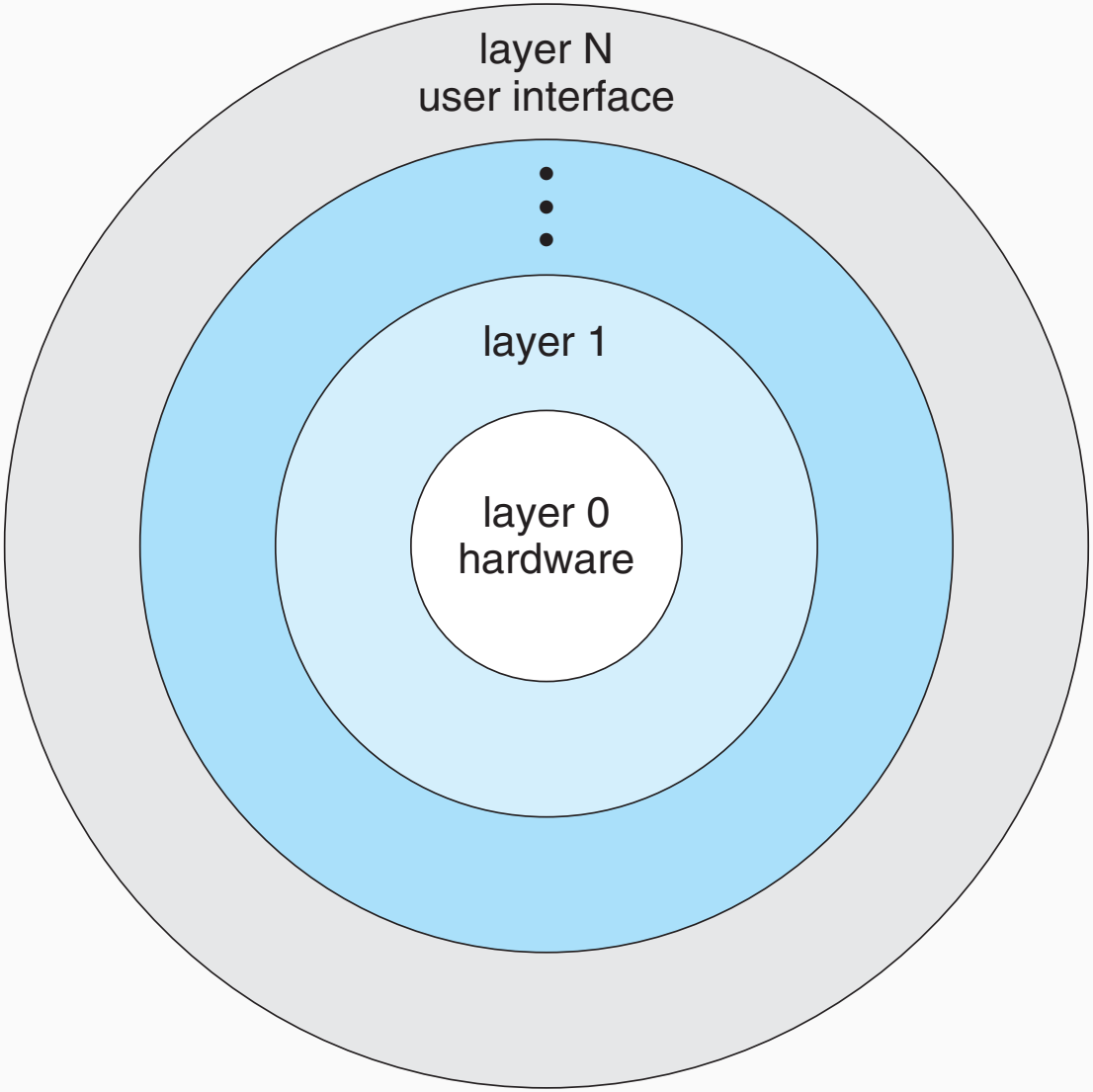

Figure 2.13 A layered operating system.

Microkernels

Este método estructura el sistema operativo eliminando todos los componentes no esenciales del kernel e implementandolos como programas del sistema y del nivel de usuario. Como resultado se obtiene un kernel mas pequeño.

- Normalmente los Microkernels proporcionan una gestión de la memoria y de los procesos mínima, además de un mecanismo de comunicación entre procesos utilizando paso de mensajes.

Microkernels

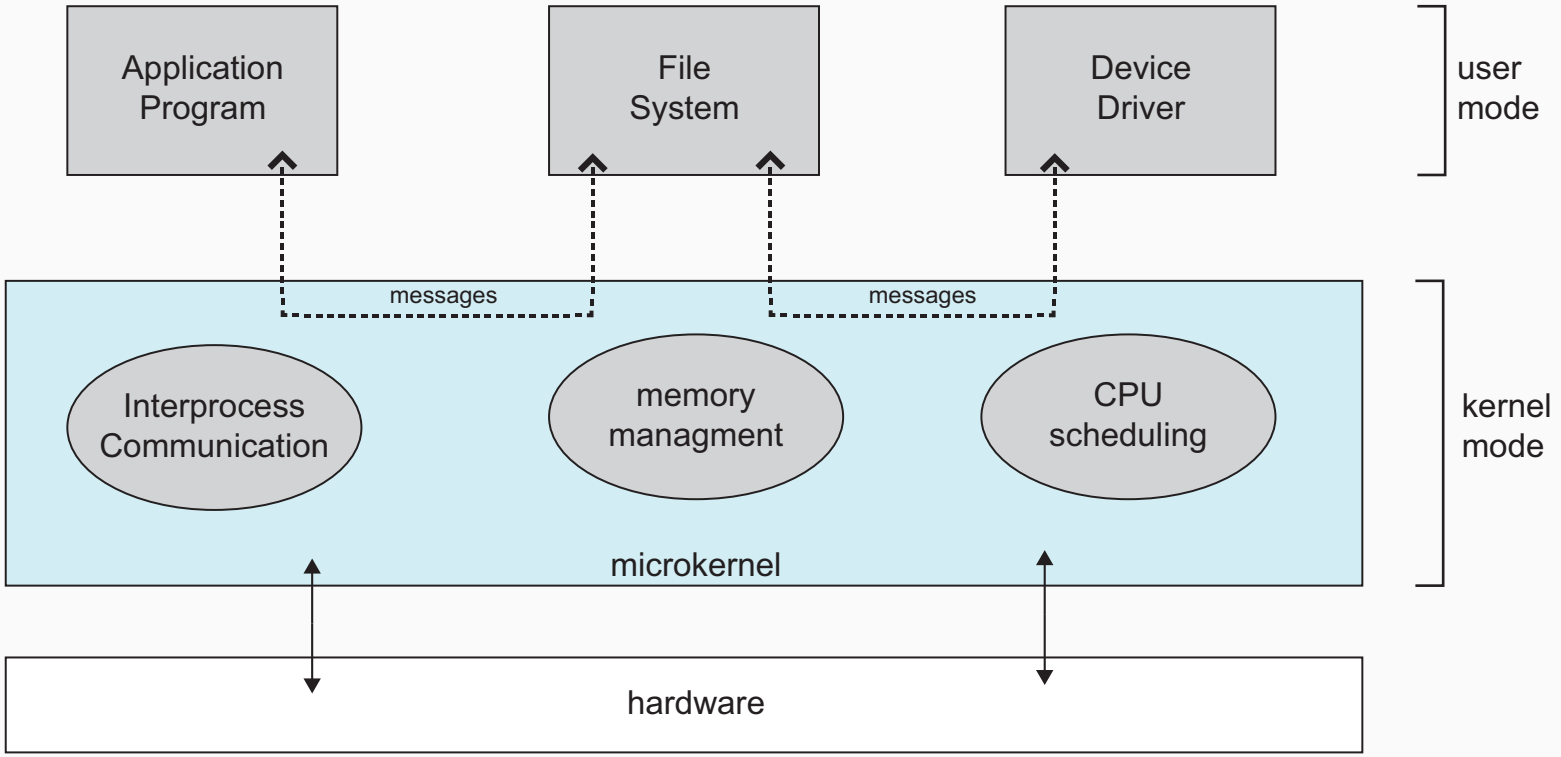

Figure 2.14 Architecture of a typical microkernel.

Figure 8:

Microkernels

Una ventaja de los Microkernels es la facilidad para ampliar el sistema operativo.

- Todos los servicios nuevos se agregan como programas de usuario y no se requiere modificar el kernel.
- Además por ser mas pequeño es mas fácil de transportar a otro sistema.
- También el Microkernels es mas seguro y fiable, dado que la mayor parte de los servicios se ejecutan como procesos de usuario.
- Si un servicio falla, el resto el sistema no se verá afectado.

Microkernels

Los sistemas de Microkernel pueden tener un rendimiento peor que otras soluciones, debido a la carga de procesamiento adicional impuesta por las funciones del sistema (paso de mensajes).

Módulos

En este caso el kernel dispone de un conjunto de componentes fundamentales y enlaza dinámicamente los servicios adicionales, bien durante el arranque o en tiempo de ejecución.

- Un diseño así permite al kernel proporcionar servicios básicos y también permite implementar ciertas características dinámicamente.
- Por ejemplo, se pueden añadir al kernel controladores de bus y de dispositivos para hardware específico y puede agregarse como módulo cargable el soporte para diferentes sistemas de archivos.

Módulos

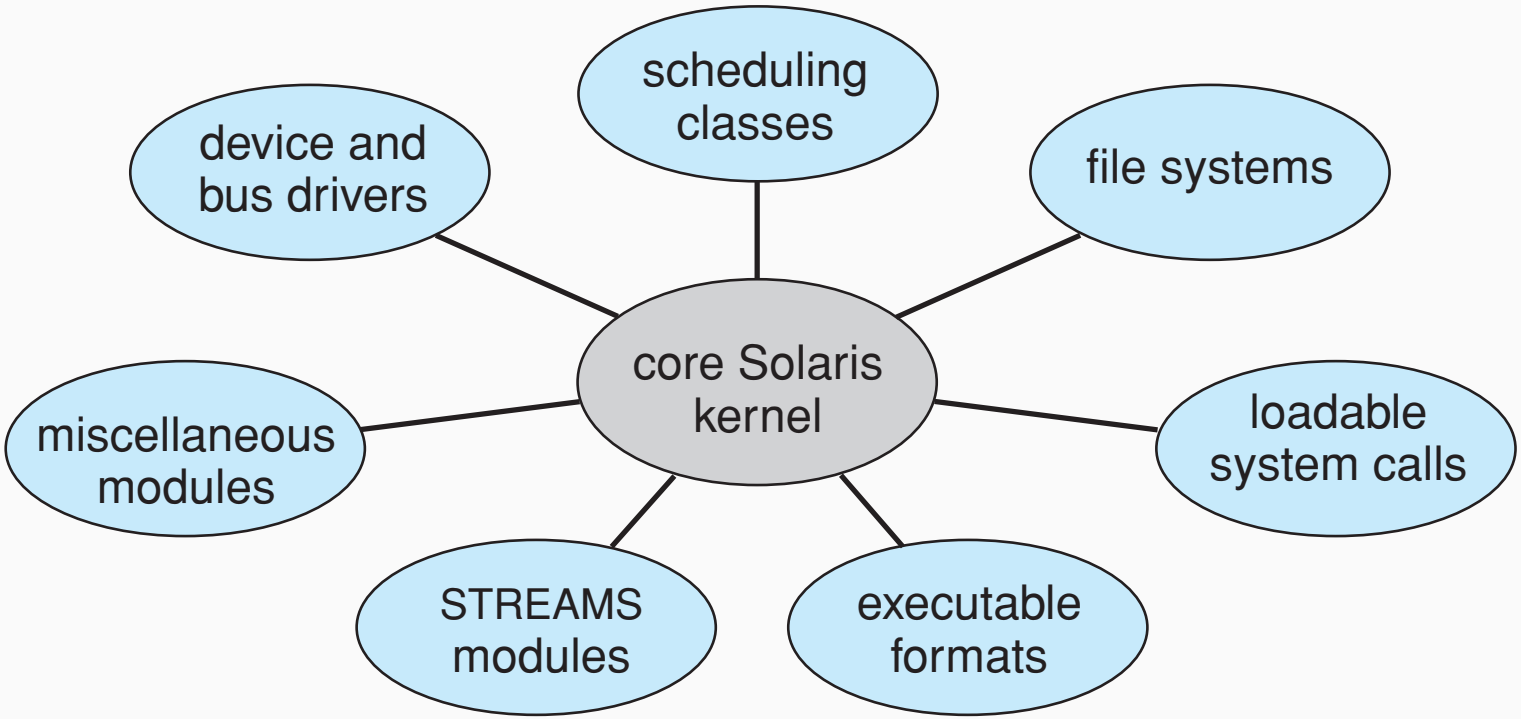

Figure 2.15 Solaris loadable modules.

Figure 9:

Módulos

El resultado es similar a un sistema de niveles, en el sentido de que cada sección de kernel tiene interfaces bien definidas y protegidas, pero es más flexible que este, porque cualquier módulo puede llamar a cualquier otro módulo.

- Además, es similar a utilizar un microkernel, ya que el módulo principal sólo dispone de las funciones esenciales; sin embargo, es más eficiente que este ya que los módulos no necesitan invocar un mecanismo de paso de mensajes para comunicarse.

Referencia bibliográfica

A.SILBERSCHATZ, P. GALVIN, y G. GAGNE, Operating Systems Concepts, 9a Edición, Cap. 2, John Wiley, 2013.