# Sistemas Operativos. Conceptos Generales

## Tabla de Contenidos




Autor: Lina García Cabrera




## 1 Introducción

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**

http://computer.howstuffworks.com/computer-hardware-pictures.htm

**Un computador es una máquina** que permite procesar la información de forma rápida y automática. Sin embargo, la utilización de un computador no es una tarea sencilla, ya que el modo en que podemos comunicarnos con él, es decir, su interfaz, es extraña y compleja al pensamiento humano.

Se entiende por **interfaz** de un objeto la parte del objeto accesible desde su exterior, que nos permite utilizarlo y consultar su estado interno.

Por ejemplo, un reloj digital presenta una interfaz constituida por una serie de botones que hacen posible la modificación su comportamiento, cambiando la hora o programando una alarma, y una pantalla y un dispositivo sonoro por el que se puede consultar su estado actual (consultar la hora u oír la expiración de una alarma). Para usar un reloj digital no es preciso conocer su funcionamiento interno, sólo hay que saber utilizar su interfaz.

La **interfaz de un computador** viene determinada por **un conjunto, normalmente pequeño, de instrucciones máquina** que permiten utilizar los dispositivos físicos o hardware (CPU, memoria y periféricos) de los que se compone.

**Si los usuarios de un ordenador tuvieran que utilizar el *hardware* a través de sus instrucciones máquina se escribirían muy pocos programas**, y estos no podrían resolver tareas excesivamente complejas, pues el uso directo de los dispositivos físicos del ordenador mediante estas instrucciones máquina es complejo, tedioso, y está lleno de detalles.

La solución que se ha ido adoptando con el tiempo para salvar esta complejidad es la de escribir capas o niveles de software.

- Una capa de software de nivel i es un conjunto de programas que trabajan directamente con la interfaz de su nivel inferior (i - 1), y presentan a su nivel superior (i + 1) una interfaz que permite utilizar el nivel i -1 de una forma más sencilla.
- Se dice que una capa software abstrae a su nivel superior de los detalles del nivel inferior, siendo, por tanto, un mecanismo de abstracción. Una capa de nivel i puede permitir a su capa superior utilizar las interfaces de sus niveles inferiores (i -1, i - 2, ...), o puede obligar a utilizar solamente la interfaz de la capa i, asegurándose así la utilización directa del nivel inferior.

El **sistema operativo** es **la capa de software más importante de un sistema informático**.

Un **sistema operativo** es el software que actúa como un intermediario entre el usuario de una computadora y el hardware de la misma. El propósito de un sistema operativo es proporcionar un entorno en el que el usuario pueda ejecutar programas de manera práctica y eficiente.

**OBJETIVOS**

- Proporcionar una visión general de los componentes de un sistema operativo.
- Proporcionar una panorámica sobre la organización básica de un sistema informático.
- Describir lo servicios que un sistema operativo proporciona a los usuarios, programas y a otros sistemas.



## 2 Definición de Sistema Operativo

teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**
Se puede definir un **sistema operativo** como un conjunto de programas que controlan directamente los recursos hardware o físicos de un ordenador (CPU, memoria principal y periféricos) proporcionando **una máquina virtual más fácil de utilizar que el hardware subyacente**. El sistema operativo es la capa de software más baja de un ordenador, como se refleja en la figura 1.

Figura 1. Niveles hardware y software de un ordenador.

En la Figura 1 se observa cómo el **sistema operativo es la única capa que trabaja directamente con el *hardware***.

Por encima del sistema operativo se encuentra un nivel formado por traductores, editores de texto e intérpretes de órdenes. Los dos primeros tipos de programas, junto con enlazadores y depuradores, son útiles para crear un nivel de abstracción cómodo para el desarrollo de programas, la utilidad de los intérpretes de órdenes se verá en un apartado posterior. La unión de los programas de las dos capas intermedias de la Figura 1 conforman el **software de sistemas** de un computador.

Por último, está el nivel constituido por los **programas de aplicación**, estos programas no dan servicio a otros programas, su finalidad es resolver problemas concretos. Son los programas que suele ejecutar un usuario no informático. Pertenecen a esta capa los procesadores de texto, hojas de cálculo, agendas electrónicas, juegos, etc.

En general, puede decirse que los sistemas operativos realizan dos funciones:

1. a) Constitución de una máquina virtual o extendida y
2. b) Utilización compartida de los recursos.

## 2.1 Constitución de una máquina virtual

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**
El sistema operativo pone al servicio del usuario una **máquina virtual** cuyas características son distintas (y más fáciles de abordar) que las de la máquina real subyacente. Algunas áreas en las que es frecuente que la máquina virtual difiera de la máquina real que la soporta son:

a. **Entrada/salida (E/S)**. La capacidad de E/S de un *hardware* básico puede que sea extremadamente compleja y que requiera sofisticados programas para su utilización. Un sistema operativo evita al usuario el problema de tener que comprender el funcionamiento de este *hardware*, poniendo a su alcance una máquina virtual mucho más sencilla de usar.
b. **Memoria**. Muchos sistemas operativos presentan la imagen de una máquina virtual cuya memoria difiere en tamaño de la de la máquina real subyacente. Así, por ejemplo, un sistema operativo puede emplear memoria secundaria (discos magnéticos) para crear la ilusión de una memoria principal mucho más extensa de la que se dispone en realidad. Alternativamente, puede repartir la memoria principal entre varios usuarios, de forma que cada uno de ellos "vea" una máquina virtual cuya memoria sea menor que la de la máquina real.

c. **Sistema de ficheros**. La mayoría de las máquinas virtuales incluyen un sistema de ficheros para el almacenamiento a largo plazo tanto de programas como de datos. El sistema de ficheros está basado en la capacidad de almacenamiento sobre disco de la máquina real. El sistema operativo, sin embargo, permite al usuario acceder a la información almacenada a través de nombres simbólicos en lugar de hacerlo a través de su posición física en el medio de almacenamiento.
d. **Protección y tratamiento de errores**. Desde el momento en que la mayoría de los ordenadores son compartidos por un determinado número de usuarios, es esencial que cada uno de ellos esté protegido de los efectos de los errores o de la mala fe de los demás. Los ordenadores varían considerablemente en lo que respecta al grado de protección que proporciona su *hardware* básico, siendo misión del sistema operativo el constituir una máquina virtual en la que ningún usuario pueda afectar de manera negativa al trabajo de los demás.
e. **Interacción a nivel de programa**. Una máquina virtual puede posibilitar la interacción entre distintos programas de los usuarios de forma que, por ejemplo, la salida de uno de ellos se emplee como entrada de otro.

**La naturaleza concreta de una máquina virtual dependerá de la aplicación particular a la que se dedique**.

> Así, por ejemplo, las características de una máquina virtual que controle un sistema de tiempo real serán distintas de las de una máquina virtual que se utilice para el desarrollo de programas.

## 2.2 Utilización compartida de recursos / Administrador de recursos

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**
**Un sistema operativo debe lograr que se compartan los recursos de un ordenador entre un cierto número de usuarios que trabajen de forma simultánea**.

Un sistema informático tiene muchos recursos:

- tiempo de CPU,
- espacio de memoria,
- espacio de almacenamiento de ficheros,
- dispositivos de E/S, etc.

El sistema operativo ***asigna y administra recursos*** **para incrementar la disponibilidad del ordenador con respecto a sus usuarios** y, al mismo tiempo, ***maximizar la utilización de los recursos***. La importancia de la utilización eficiente de estos recursos depende de su coste.

De igual modo un sistema operativo **es un *programa de control* que gestiona la ejecución de los programas de usuario para evitar errores y mejorar el uso de la computadora**. Debe decidir cómo asignar los recursos a los programas de modo que la computadora pueda operar de forma eficiente y equitativa.

El sistema operativo dirige el procesador en el uso de los otros recursos del sistema y en la ejecución de otros programas.

Además, lleva la cuenta de quién utiliza los recursos para distribuirlos, debe contabilizar su utilización y decidir la concesión de los recursos a diferentes programas o usuarios en conflicto.

> ***El mecanismo de control que ejerce el SO es inusual***. No es algo externo, el sistema operativo funciona del mismo modo que el resto del software (es un **programa que el procesador ejecuta**). Por tanto, si se quiere que los demás programas lleguen a ejecutarse, el **sistema operativo debe ceder el control con frecuencia y depende del procesador para recuperarlo**.

## 3 Sistema Operativo de Propósito General

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**

Para poder comprender mejor la estructura de un sistema operativo necesitas algunos conocimientos de arquitectura y organización de un sistema informático. Por favor, antes de leer esto lea sobre estos temas.

Una de los aspectos más importantes de los sistemas operativos es la capacidad de multiprogramar. Un programa de usuario no puede mantener la CPU o los dispositivos de E/S ocupados continuamente, a estos sistemas se les conoce como **sistemas de flujo único o monoprogramados**.

Un **sistema de flujo único o monogramado** dedica la máquina por completo a la ejecución de una sola tarea, no importa lo larga o lo corta que sea.

La multiprogramación incrementa el uso de la CPU organizando las tareas de modo que la CPU siempre tenga algo que ejecutar. Para ello, mantiene varios trabajos en memoria al mismo tiempo. La **multiprogramación** consiste en **la ejecución simultánea de varios programas que residen en la memoria principal**, dividiendo el procesador central su tiempo entre ellos de acuerdo con los recursos (tales como canales o dispositivos) que necesite en cada momento cada uno de ellos. Cuando un programa tiene que esperar a que se realice alguna tarea (por ejemplo, completar una operación de E/S),

1. el sistema operativo elige otro programa,
2. la CPU conmuta a ese otro trabajo y se ejecuta.

Cuando un trabajo deja de esperar porque ha completado la tarea que tenía que hacer,

1. llegará un momento en el que el sistema operativo lo selecciona y
2. vuelve obtener la CPU.

Mientras haya al menos un trabajo que necesite ejecutarse, la CPU nunca estará inactiva.

De esta forma es posible, teniendo almacenado un conjunto adecuado de tareas en cada momento, obtener una utilización óptima de los recursos disponibles. Ello incluye la utilización del procesador central, ya que en tanto que una tarea esté esperando el final de una transferencia de E/S, este procesador puede pasar a trabajar en alguna otra tarea que esté pendiente en la máquina. La carga que recae sobre el sistema operativo consiste en el control de los recursos, así como la protección de cada tarea frente a las actividades de las otras. Un sistema operativo de este tipo recibe el nombre de **monitor de batch de varios flujos**.

Los *sistemas multiprogramados* proporcionan un entorno en el que se usan eficazmente los recursos del sistema (CPU, memoria y periféricos) pero *NO ofrecen la interacción del usuario* con el sistemas informático. El **tiempo compartido** es una extensión de la multiprogramación. Los ***sistemas de tiempo compartido son multiprogramados*** (la CPU ejecuta varios trabajos conmutando entre ellos) pero ahora las conmutaciones no sólo suceden cuando los programas no pueden progresar en su ejecución sino que ***el tiempo de CPU se reparte entre los distintos programas***. Las conmutaciones se producen tan frecuentemente que los usuarios pueden interactuar con cada programa mientras se está en ejecución. De este modo el usuario tiene la ilusión de que todo el computador se le dedica exclusivamente a él, al recibir unos tiempos de respuesta aceptables.
El tiempo compartido permite **sistemas informáticos interactivos**. De igual modo, permite que muchos usuarios compartan simultáneamente la computadora, por tanto, **sistemas informáticos multiusuario**.
Un sistema de tiempo compartido emplea los mecanismos de multiprogramación y de planificación de la CPU para proporcionar a cada usuario una pequeña parte de una computadora de tiempo compartido. Cada usuario tiene al menos un programa distinto en memoria. ***Un programa en memoria y en ejecución se denomina proceso***. Cuando se ejecuta un proceso, normalmente sólo se ejecuta un período de tiempo breve, antes de terminar o de necesitar una operación de E/S. La E/S puede ser interactiva, es decir, la salida va a la pantalla y la entrada procede el teclado o del ratón. La E/S interactiva se realiza a la velocidad a la que trabajan las personas, una velocidad muy lenta para una computadora. En lugar de dejar a la CPU sin hacer nada mientras se realiza la entrada, el sistema operativo hará que la CPU conmute al programa de otro usuario.

El tiempo compartido y, por tanto, la multiprogramación requieren mantener simultáneamente en memoria trabajos. Como la memoria principal suele ser pequeña para todos los trabajos, éstos se mantienen inicialmente en el disco, en una **cola de trabajos**. El planificador de trabajos decide qué trabajos se llevan a memoria principal para ser ejecutados.

Tener varios programas en memoria principal al mismo tiempo requiere algún mecanismo de gestión de memoria. Si hay varios trabajos preparados en memoria principal, el sistema debe elegir entre ellos, esta decisión se denomina **planificación de la CPU o a corto plazo**.

En un sistema de tiempo compartido, el sistema operativo debe garantizar un tiempo de respuesta razonable. Esto se consigue a veces con un mecanismo de **intercambio** entre memoria y disco. También se puede conseguir este objetivo con la **memoria virtual**, una técnica que permite ejecutar un proceso que no está totalmente en memoria principal. Este esquema de memoria virtual permite a los usuarios ejecutar programas que sean más grandes que la memoria física real.

Los sistemas de tiempo compartido también tienen que proporcionar un sistema de archivos que reside en una colección de discos. El sistema operativo deberá tener mecanismos para la gestión de estos discos.

## 4 Servicios de un Sistema Operativo

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**
**Un sistema operativo proporciona un entorno para la ejecución de programas**. El sistema presta servicios a los programas y a los usuarios de dichos programas.

Algunas **servicios del sistema operativo** proporciona funciones que resultan **útiles al usuario**:

- **Interfaz de usuario**. Los sistemas operativos incluyen una interfaz de usuario que puede ser de línea de órdenes que permite introducir y editar órdenes de texto. Otros también incorporan una interfaz de proceso por lotes en la que las órdenes y la directivas para controlar esas órdenes se introducen en ficheros que luego se ejecutan (en unix, los programas en *shell*). Hoy en día, todos los sistemas operativos utilizan una interfaz gráfica de usuario, un sistema de ventanas, desde el que se pueden ejecutar estas órdenes de forma explícita o implícita.
- **Ejecución de programas**. El sistema operativo carga un programa en memoria y lo ejecuta. Todo programa puede terminar su ejecución, de forma normal o anormal (indicando un error).
- **Operaciones de E/S**. Un programa en ejecución puede necesitar realizar operaciones de E/S, dirigidas a un fichero o a un dispositivo de E/S. Por motivos de eficacia y protección, los usuarios no pueden controlar de forma directa a lo dispositivos y lo hacen a través del sistema operativo.
- **Manipulación del sistema de ficheros**. Los programas necesitan leer y escribir en archivos y directorios, crearlos, borrarlos usando su nombre, realizar búsquedas en un fichero o mostrar la información que contienen. También algunos sistema operativos incluyen mecanismos de gestión de permisos para permitir o no el acceso de lectura, escritura o ejecución al fichero.
- **Comunicaciones**. Hay circunstancias en las que un proceso necesita intercambiar información con otro. Esta comunicación puede hacerse entre procesos que se están ejecutando en la misma computadora o entre procesos que se ejecutan en computadoras diferentes conectadas a través de una red. La comunicación se puede realizar utilizando memoria compartida o mediante el paso de mensajes.
- **Detección de errores**. Los errores se pueden producir en el hardware de procesador y de memoria (error de memoria o un fallo de alimentación), en un dispositivo de E/S (error de paridad, fallo en la conexión de la red, falta papel en la impresora) o en los programas de usuario (un desbordamiento aritmético, un intento de acceso a una posición de memoria ilegal o un uso excesivo de CPU). Para cada tipo de error, el sistema operativo realiza la acción apropiada para garantizar un funcionamiento correcto y coherente.

Otras funciones del **sistema operativo** garantizan la **eficacia del propio sistema**:

- **Asignación de recursos**. Con varios usuarios tendremos varios trabajos ejecutándose al mismo tiempo y cada uno necesita unos recursos. El sistema operativo gestiona estos recursos (ciclos de CPU, memoria principal, espacio en el disco, dispositivos de E/S) necesitan software especial que los gestione.
- **Registro o monitorización**. El sistema operativo hace un seguimiento del uso de los recursos por parte de los programas de usuario con el fin de tener estadísticas de uso que permitan reconfigurar el sistema y mejorar los servicios.
- **Protección y seguridad**. La ejecución de un proceso no puede interferir con la de los demás procesos o con el propio sistema operativo. La protección garantiza que todo los accesos a los recursos estén controlados. La seguridad evita los intrusos solicitando una contraseña al usuario. Esta defensa se extiende a los dispositivos de E/S (adaptadores de red) frente a intentos de acceso ilegales.

# 5 Interfaces con el Sistema Operativo

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**
Los usuarios y los programas interactúan con el sistema operativo de diversas formas para demandar sus servicios mediante un **intérprete de órdenes** (que puede ser textual o gráfico) o con **llamadas al sistema**.

## 5.1 El Intérprete de órdenes

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**
Cuando un usuario se conecta a un ordenador se inicia un **intérprete de órdenes** (en entornos UNIX llamados ***shells***).

> Por ejemplo, en los sistemas Unix y Linux, hay varios shells diferentes entre los que el usuario puede elegir tales como *shell Bourne sh*, la *shell C csh*, la *shell Korn ksh*, etc.

> El sistema operativo Mac OS X proporciona la interfaz *Aqua*.

El **intérprete de órdenes** es un programa que muestra un indicador (***prompt***) formado por algunos caracteres, que pueden incluir el directorio de trabajo [1] (un posible indicador en MS DOS es C:\>), que indica al usuario que es posible introducir una orden.

> El usuario escribirá una orden, por ejemplo
> `C:\> copy fich fich2`
> `juan@cosmos:~$ rm file.txt`
> y pulsará la tecla `return`. En un entorno UNIX, o MS DOS, la primera palabra es el nombre de un fichero (`copy` y `rm`) que contiene un programa, siendo el resto de la línea una serie de argumentos, separados por espacios, que toma dicho programa.

El intérprete de órdenes identifica el fichero, lo busca, lo carga en memoria y lo ejecuta pasándole sus argumentos.
Una excepción a esto son las **órdenes internas**, que el intérprete implementa como rutinas suyas, y que no tienen, por tanto, un programa asociado guardado en disco. El intérprete de órdenes realiza entonces una o varias llamadas al sistema para ejecutar dicho programa. Cuando el programa finalice (al realizar una llamada al sistema `exit`) el control vuelve al programa que lo lanzó (el intérprete de órdenes), mostrando éste otra vez el indicador, y repitiéndose el ciclo.

Así pues, el **intérprete de órdenes es un programa de sistema que sirve de interfaz entre el sistema operativo y un usuario, utilizándolo este último para ejecutar programas**.

A diferencia de un programador, un "usuario final" realiza todas las llamadas al sistema indirectamente, a través de las llamadas al sistema de los programas que ejecuta. En muchos sistemas se ha optado por *sustituir el intérprete de órdenes por un programa que utiliza ventanas*. En estos programas aparecen iconos que el usuario puede seleccionar mediante un ratón. Cuando el usuario selecciona un icono que representa un programa se realizan llamadas al sistema para ejecutar un fichero, asociado al icono, que contiene el programa.

Por lo tanto, se sustituye la interfaz del usuario para ejecutar programas, pero la interfaz con el sistema operativo no cambia.

Es el momento de hacer una aclaración. Existen una serie de programas muy importantes (como traductores, editores de texto, ensambladores, enlazadores e intérpretes de órdenes) que ayudan al programador a realizar sus programas, y que vienen en el lote con cualquier sistema operativo. Estos programas, que forman parte de los programas de sistema o software de sistemas, utilizan llamadas al sistema, pero no son parte del sistema operativo. El sistema operativo es el código que acepta llamadas al sistema, y realiza un procesamiento para satisfacer dichas llamadas. El sistema operativo es el programa de sistema más importante.

En las prácticas de la asignatura se verá cómo se puede escribir un *shell* simple en UNIX.

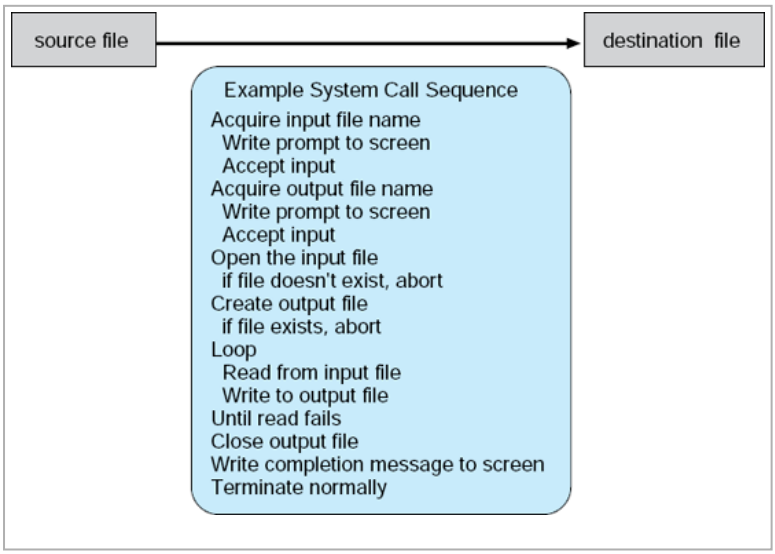
---

**[1] Directorio de trabajo** es el directorio dentro de sistema de ficheros en el que se encuentra en cada momento el usuario situado

## 5.2 Las Llamadas al Sistema

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**
Ya se ha comentado que el sistema operativo es una interfaz que oculta las peculiaridades del *hardware*. Para ello ofrece una serie de servicios que constituyen una máquina virtual más fácil de usar que el *hardware* básico. Estos servicios se solicitan mediante **llamadas al sistema**.

Las **llamadas al sistema** permiten que un programa en ejecución solicite servicios al sistema operativo.

> Ejercicio!! Tenemos un programa sencillo que lee datos de un fichero y los escribe en otro, ¿cuántas llamadas al sistema se necesitan?
>
> 
> 
>
> Figura 2. Ejemplo de cómo se usan las llamadas al sistema.

La forma en que se realiza una llamada al sistema consiste en

1. **colocar una serie de parámetros en un lugar específico** (como los registros del procesador), para después
2. **ejecutar una instrucción del lenguaje máquina** del procesador denominada **`trap`** (en castellano, trampa). La ejecución de esta instrucción máquina hace que *el hardware guarde el contador de programa y la palabra de estado del procesador* (PSW, *Processor Status Word*) en un lugar seguro de la memoria,
3. *cargándose un nuevo contador de programa y una nueva PSW*. Este nuevo contador de programa contiene una dirección de memoria donde reside una parte (un programa) del sistema operativo, el cual se encarga de llevar a cabo el servicio solicitado.
4. Cuando el sistema operativo **finaliza el servicio**, coloca un código de estado en un registro para indicar si hubo éxito o fracaso, y ejecuta una instrucción **`return from trap`**, esta instrucción provoca que el *hardware restituya el contador de programa y la PSW del programa que realizó la llamada al sistema*, prosiguiéndose así su ejecución.

Normalmente los lenguajes de alto nivel tienen una (o varias) **rutinas de biblioteca** por cada llamada al sistema (la API, *Application Programming Interface*). Dentro de estos procedimientos se aísla el código (normalmente en ensamblador) correspondiente a la carga de registros con parámetros, a la instrucción `trap`, y a obtener el código de estado a partir de un registro. La **finalidad de estos procedimientos de biblioteca es ocultar los detalles de la llamada al sistema**, ofreciendo una interfaz de llamada a procedimiento. Como una llamada al sistema depende del *hardware* (por ejemplo, del tipo de registros del procesador), la utilización de rutinas de biblioteca hace el código portable.

El número y tipo de llamadas al sistema varía de un sistema operativo a otro. Se pueden agrupar en cinco categorías:

1. control de procesos,
2. manipulación de ficheros,
3. manipulación de dispositivos,
4. mantenimiento de información y
5. comunicaciones.

Existen, por lo general, llamadas al sistema para ejecutar ficheros que contienen programas, pedir más memoria dinámica para un programa, realizar labores de E/S (como la lectura de un carácter de un terminal), crear un directorio, etc.

Ejemplos de rutinas de biblioteca que realizan llamadas al sistema en un entorno C-UNIX son: `read`, `write`, `malloc`, `exec`, etc.

> **Ejemplo de la biblioteca estándar C**
> Un programa en C invoca `printf()`. La biblioteca C intercepta esta solicitud e invoca las llamadas al sistema necesarias del sistema operativo, en este caso, la llamada al sistema `write()`. La biblioteca C toma el valor devuelto por `write()` y se lo pasa al programa de usuario.

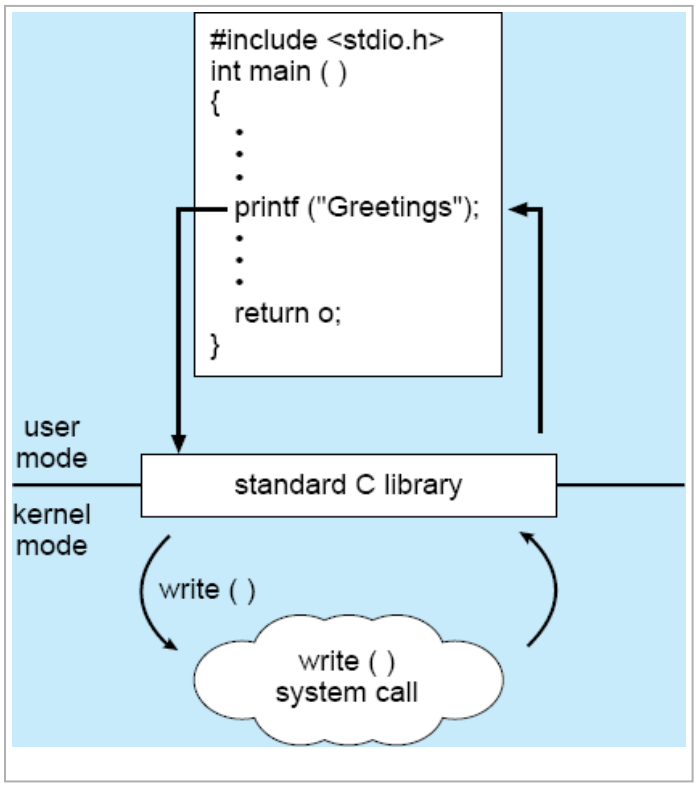



Figura 3. Tratamiento del procedimiento `write ()` de la biblioteca estándar C [Silberschatz, 2010].

## 6 Llamadas al Sistema y Protección

2º Grado en Ingeniería en Informática
teoría de Sistemas Operativos
**Bloque Sistemas Operativos: Conceptos Generales**
En los sistemas de multiprogramación se pueden ejecutar varios programas a la vez. Cuando se ejecutan simultáneamente varios programas (probablemente pertenecientes a distintos usuarios) hay que proteger a un programa de la acción de los demás.

> Por ejemplo, no se debe permitir que un programa pueda modificar posiciones de memoria principal donde se almacena otro programa, ni tampoco que un usuario borre los ficheros de otro usuario.

Para llevar a cabo esta protección, el **sistema operativo se apoya en varios mecanismos proporcionados por el** *hardware*. Los mecanismos de protección de memoria se verán en los temas de gestión de memoria. Ahora vamos a comentar el **modo supervisor** (o **modo núcleo**) y el **modo usuario**. La mayoría de los procesadores tienen ***dos modos de funcionamiento llamados modo supervisor y modo usuario*** (un bit de la PSW suele indicar el modo de funcionamiento). En **modo supervisor** está permitido la ejecución de cualquier instrucción máquina, sin embargo, en **modo usuario** no se permite la ejecución de algunas instrucciones reservadas que llevan a cabo funciones tales como:

a. autorizar e inhibir las interrupciones.
b. conmutar un procesador entre distintos procesos (un proceso es un programa en ejecución, este concepto se verá en el tema procesos).
c. acceder a los registros utilizados por el hardware de protección de la memoria.
d. realizar operaciones de E/S.
e. parar la CPU.

**El sistema operativo se ejecuta en modo supervisor con acceso total al hardware, mientras que los programas de usuario se ejecutan en modo usuario**. Cuando un programa necesita utilizar un recurso *hardware*, por ejemplo, quiere realizar una operación de E/S, debe realizar una llamada al sistema para que el sistema operativo lleve a cabo la operación por él. ***La realización de una llamada al sistema conmuta automáticamente la CPU a modo supervisor***.

> Así se garantiza que, por ejemplo, un usuario al que no se le ha concedido una impresora utilice instrucciones máquina para acceder directamente al controlador de la impresora, interfiriendo con el programa al que se le concedió la impresora.

La vuelta al modo usuario desde el modo supervisor se lleva a cabo mediante una instrucción, también reservada.

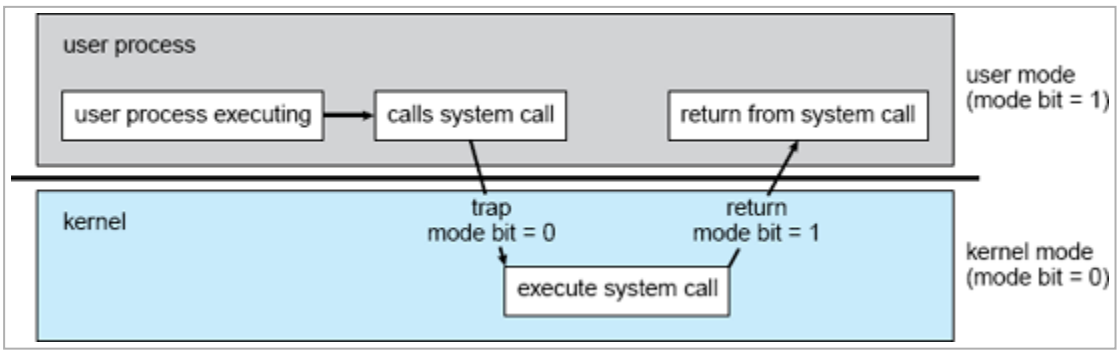



Figura 4. Transición del modo usuario al modo kernel [Silberschatz, 2010].