1. Componentes

1.1. Evolución histórica de los sistemas operativos

La evolución de los sistemas operativos se suele dividir en generaciones, correspondientes a diferentes hitos tecnológicos. Así, tenemos:

1.ª generación (1945-1955): los primeros ordenadores estaban basados en **tubos de vacío**. **No existían los sistemas operativos**. Se programaba directamente en código máquina, generalmente a través de paneles de cableado combinables de distintos modos. A finales de 1950, las tarjetas perforadas (unas cartulinas agujereadas que codificaban instrucciones y datos) sustituyen a los paneles enchufables.

2.ª generación (1955-1965): aparecen los **transistores**, que permiten construir ordenadores más pequeños y potentes. Los programas se procesaban en lotes —es decir, cuando se cargaba un proceso ,este tenía a su disposición todos los recursos de la máquina hasta finalizar la tarea— y existía un programa de control que interpretaba las tarjetas. Este programa de control constituye un antecedente de los actuales intérpretes de comandos. La programación se llevaba a cabo en tarjetas perforadas (ensamblador o FORTRAN).

3.ª generación (1965-1980): los circuitos integrados irrumpen en la electrónica, consiguiendo de nuevo fabricar ordenadores más pequeños, potentes, eficientes y económicos. Aparecen asimismo los sistemas operativos de tiempo compartido y la multiprogramación (gracias a la cual, un ordenador es capaz de hacer más de una cosa, aunque no simultáneamente). Surgen lenguajes de programación como BASIC o PASCAL.

4.ª generación (1980-actualidad): los circuitos integrados se generalizan a gran escala y aparecen los ordenadores personales. Surgen **los sistemas operativos** de tiempo real y la multitarea (que permite a un ordenador realizar más de una labor de forma simultánea). Tenemos, además, equipos con múltiples procesadores, sistemas operativos en red, distribuidos, etcétera. Muchos de los sistemas operativos que usamos actualmente aparecieron en esta generación: Microsoft Windows, Mac OS y GNU/Linux.

1.2. Elementos

Los elementos que conforman los sistemas operativos son:

El **núcleo**, también llamado *kernel*, ubicado en los cimientos del sistema operativo y en contacto directo con el hardware. Controla las interrupciones, la asignación de trabajos al procesador o la comunicación entre los diferentes programas. En general, se encarga de coordinar al resto de elementos del sistema . Por su importancia dentro del núcleo, destacaremos la tarea del **planificador**, encargado de asignar el tiempo de procesador que corresponde a cada proceso en el caso de los programas en ejecución. Aunque existen diferentes mecanismos de planificación, prácticamente todos asignan un tiempo máximo a cada proceso. Cuando dicho proceso no puede terminarse en ese tiempo asignado, vuelve a la cola de procesos preparados y se le asigna tiempo a otro proceso distinto.

El **administrador de memoria**, gestiona la memoria principal del sistema (RAM) y la asigna en porciones a los diferentes procesos según sus necesidades. Los sistemas operativos actuales aplican la técnica de la **memoria virtual para gestionar la memoria física. Esta técnica permite un mayor aprovechamiento de la memoria principal**. Además, el administrador de memoria también gestiona qué parte de la información que requiere el proceso se carga en la memoria principal y cuál se queda en la memoria secundaria, así como el movimiento de una memoria a otra (*swapping*), que se produce cuando la memoria principal está llena y cuando un proceso necesita más memoria para su ejecución.

La gestión del **sistema de entrada y salida** se encarga de presentar al usuario o de guardar los datos con independencia del dispositivo de entrada o salida que se esté utilizando. Así, el usuario recibe/almacena información en un dispositivo de una misma manera, y es el sistema operativo el que adapta esa petición o esos datos al funcionamiento particular del dispositivo concreto. En muchos de estos dispositivos de entrada/salida se usa un ***spool*** —o dispositivo de almacenamiento masivo—, el cual va almacenando toda la información que se irá enviando al dispositivo físico de entrada/salida según vaya estando disponible. De esta forma se garantiza que ningún proceso se quede bloqueado a la espera de que el dispositivo de entrada/salida esté disponible.

Por otro lado, tenemos el **administrador de archivos,** que es el responsable de la estructura de los datos y programas de cada uno de los usuarios. Vigila la creación, la modificación y el borrado de archivos y carpetas. Otra función importante del mismo es la gestión de los **permisos y privilegios** de los usuarios sobre los archivos o carpetas, de manera que solo aquellos usuarios que están autorizados puedan hacer uso de los mismos.

Algunas veces se considera al **intérprete de comandos** un elemento más de los sistemas operativos, pues funciona como una interfaz entre el usuario y el sistema, y es el encargado de interpretar y traducir las ordenes del usuario para que sean procesadas por el sistema.

1.3 Estructura

Monolítica: es la que presentaban los primeros sistemas operativos. En realidad, equivale a carecer de estructura. Consistía en un solo programa con rutinas que se podían llamar entre sí. Estas estructuras eran muy difíciles de mantener y, casi siempre, se fabricaban expresamente a la medida del sistema en el que se iban a ejecutar.

Jerárquica o por **capas**. Según los usuarios se fueron volviendo más exigentes, los sistemas operativos incrementaron también su complejidad y funcionalidad. Por estos motivos, y para facilitar el mantenimiento de los equipos, se fragmentó el sistema operativo en partes más pequeñas, con funciones claramente diferenciadas y que ofrecían una interfaz para operar con las capas o módulos adyacentes.

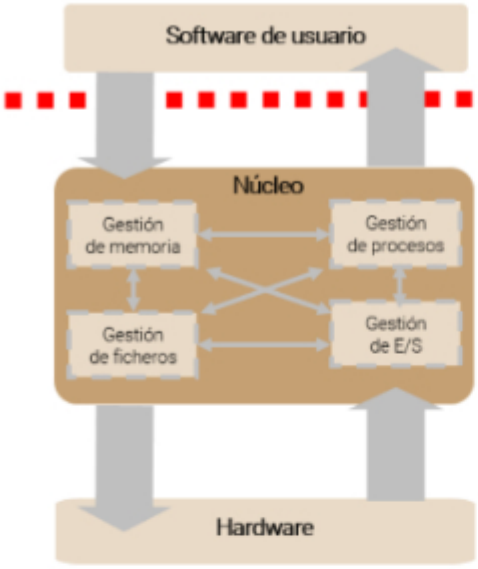

Figura 1.1. Estructura monolítica.

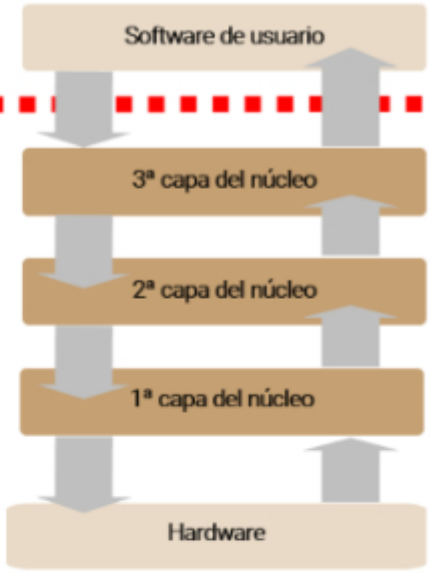

Figura 1.2. Estructura por capas.

El modelo **microkernel** —o **cliente/servidor**— organiza las funciones del sistema operativo en módulos sencillos y aislados del núcleo. Gracias a ello, tenemos módulos para la gestión de entrada/salida, la memoria, el sistema de archivos, etc. Esta estructura ofrece una mayor tolerancia frente a los fallos, una mejor seguridad y más portabilidad entre el hardware de diferentes sistemas.

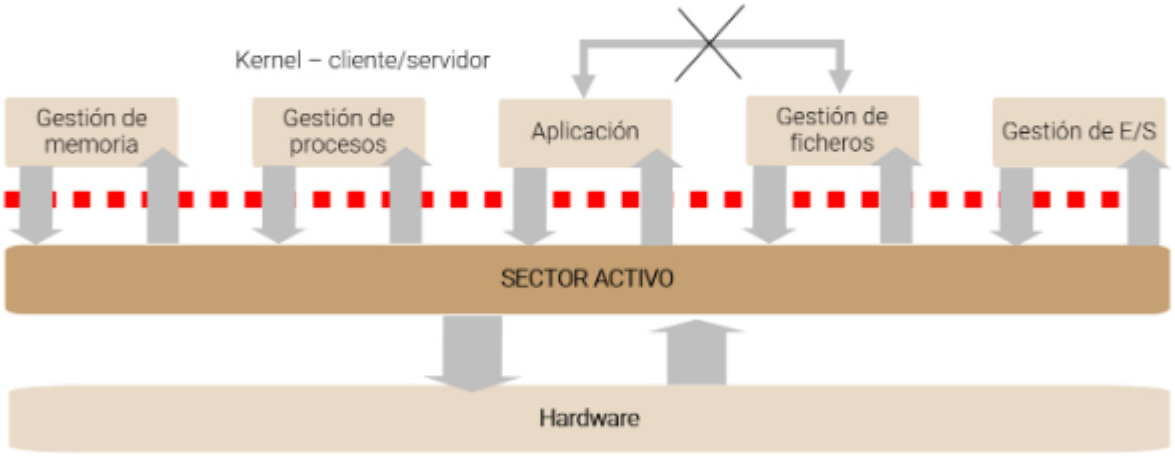

Figura 1.3. Estructura microkernel o cliente/servidor.