UNIVERSITAT DE
BARCELONA

Sistemas Operativos I
Marco histórico y conceptual

Autor: **Oliver Díaz Montesdeoca**
Fecha: Febrero 2022

Contents

1 SISTEMAS OPERATIVOS: MARCO HISTÓRICO Y CONCEPTUAL

En este capítulo se presenta el marco conceptual donde se sitúa la asignatura de Sistemas Operativos I. En el apartado 1.1 se introduce brevemente el concepto de sistemas operativos. El apartado 1.2 repasa brevemente la historia y evolucin de los sistemas operavitos. El apartado 1.3 presenta una taxonomía de los principales paradigmas de los sistemas operativos y finalmente el apartado 4 resume y concluye el capítulo.

1.1 Introducción

El sistema operativo es una capa de software que actúa como intermediario entre el usuario de una máquina (p.ej. ordenador) y el hardware de la misma (figura 1). El propósito de un sistema operativo es proporcionar un entorno en el que un usuario pueda ejecutar programas en una manera fiable, eficiente y segura.

El hardware debe proporcionar los mecanismos adecuados para garantizar la correcto funcionamiento del sistema informático y para evitar que los programas interfieran con el correcto funcionamiento del sistema. Los sistemas operativos realizan tareas básicas, como reconocer la entrada del teclado, enviar la salida a la pantalla de visualización, realizar un seguimiento de los archivos y directorios en el disco y controlar dispositivos periféricos como unidades de disco e impresoras. Internamente, los sistemas operativos varían mucho en su diseño y se pueden encontrar en un gran número de dispositivos electrónicos, incluidos ordenadores, teléfonos móviles, cajeros automáticos e incluso muchos automóviles modernos. Aún así, la mayoría de los usuarios estan familiarizados con las ofertas de sistemas operativos de Microsoft (Windows), Apple (OS X) y, en menor medida, Linux / GNU (Ubuntu, Fedora, etc.).

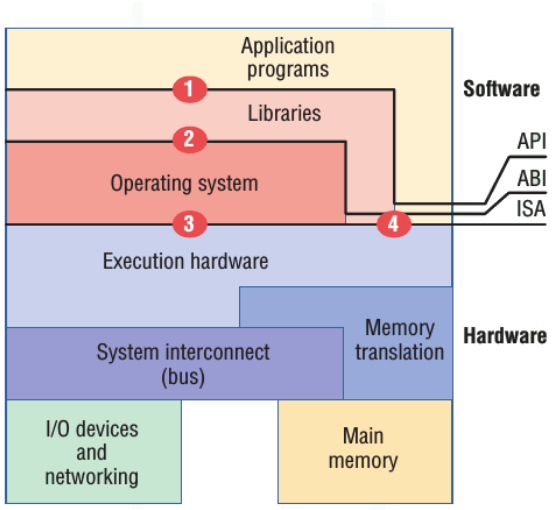

Figure 1: Interfícies de una máquina.

1.2 Breve historia de los sistemas operativos

El primer ordenador digital fue diseñado por el matemático inglés Charles Babbage (1792-1871). Aunque Babbage pasó la mayor parte de su vida y su fortuna tratando de construir su "motor analítico", nunca logró que funcionara correctamente porque era puramente mecánico y la tecnología de su época no podía producir las ruedas, engranajes y engranajes necesarios para la alta precisión que necesitaba. Como dato histórico, Babbage se dio cuenta de que necesitaría software para su motor analítico, por lo que contrató a una joven llamada Ada Lovelace, que era hija del famoso poeta británico Lord Byron, como la primera programadora del mundo. El lenguaje de programación Ada lleva su nombre.

Primera generación (1945-1955). Después de los infructuosos esfuerzos de Babbage, se avanzó poco en la construcción de computadoras digitales hasta la Segunda Guerra Mundial. A mediados de la década de 1940, Howard Aiken en Harvard, John von Neumann en el Instituto de Estudios Avanzados de Princeton, J. Presper Eckert y William Mauchley en la Universidad de Pensilvania y Konrad Zuse en Alemania, entre otros, lograron construir motores de cálculo. Los primeros usaban relés mecánicos pero eran muy lentos así que éstos fueron reemplazados por tubos de vacío. Estas máquinas eran enormes y llenaban habitaciones enteras con decenas de miles de tubos de vacío, pero seguían siendo millones de veces más lentas que incluso las computadoras personales más baratas disponibles en la actualidad. Una de los primeros ordenadores de uso generas fué el ENIAC (Electronic Numerical Integrator and Computer), que tuvo un uso militar durante la II Guerra Mundial (figura 2).

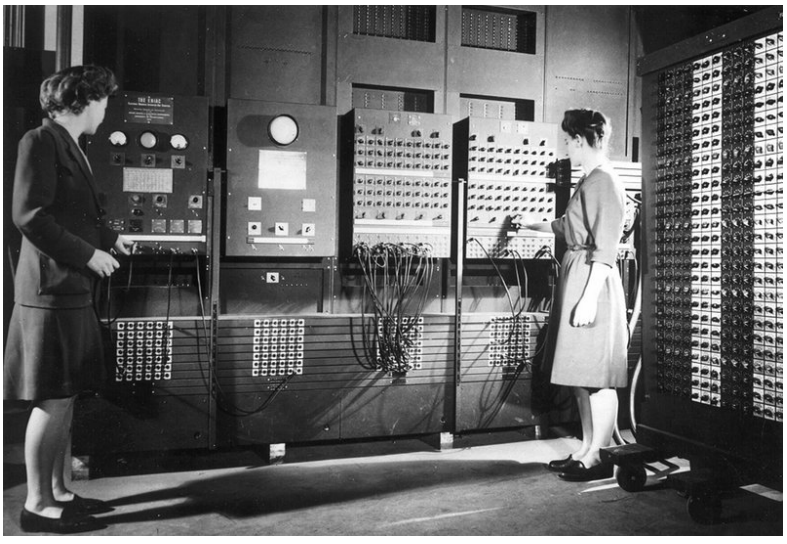

Figure 2: Programadoras configurando ENIAC en 1946. Fuente: `https://www.npr.org/sections/alltechconsidered/2014/10/06/345799830/the-forgotten-female-programmers-who-created-modern-tech`.

En estos primeros días, la constucción, programación y mantenimiendo de cada máquina dependia únicamente de un grupo de personas. En esos años no existía ni siquiera el concepto de sistema operativo y todo intento de programación se realizó en lenguaje de máquina, a menudo conectando

tableros de conexiones para controlar las funciones básicas de la máquina. Prácticamente todos los problemas eran cálculos numéricos sencillos, como crear tablas de senos, cosenos y logaritmos.

Segunda generación (1955-1965). A principios de la década de 1950 la rutina había mejorado un poco con la introducción de tarjetas perforadas y se introdujo el **primer sistema operativo**, que se llamó **GMOS** y fue creado por General Motors para la máquina 701 de IBM. Los sistemas operativos en la década de 1950 se llamaban sistemas de procesamiento por lotes de flujo único porque los datos se enviaban en grupos. Estas nuevas máquinas se llamaron **pmainframes** y fueron utilizadas por operadores profesionales en grandes salas de computación. Los ordenadores se volvieron lo suficientemente fiables como para poder fabricarlos y venderlos. Solo las grandes empresas o las principales agencias gubernamentales o universidades podían permitirse el precio multimillonario. Por primera vez, hubo una clara separación entre diseñadores, constructores, operadores, programadores y personal de mantenimiento.

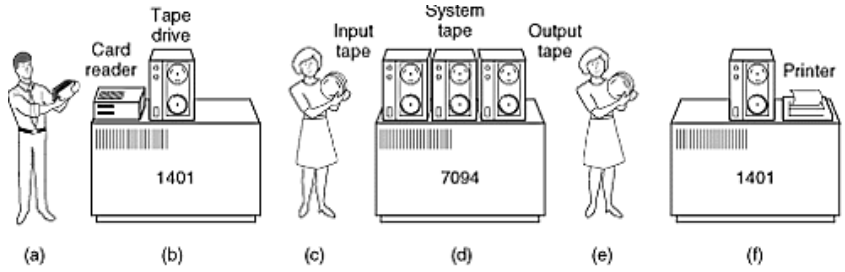

Figure 3: Sistema de lotes temprano. (a) Los programadores traen tarjetas a 1401. (b) 1401 lee lotes de trabajos en cinta. (c) El operador lleva la cinta de entrada a 7094. (d) 7094 realiza los cálculos. (e) El operador lleva la cinta de salida a 1401. (f) 1401 imprime la salida. Figura extraída de [1].

Se comenzó a utilizar diferentes tarjetas perforadas (JOB, FORTRAN, LOAD, END) para controlar los diferentes procesos. Estas tarjetas de control primitivas fueron las precursoras de los lenguajes de control de trabajos e intérpretes de comandos modernos.

Las computadoras grandes de segunda generación se utilizaron principalmente para cálculos científicos y de ingeniería, como resolver las ecuaciones diferenciales parciales que a menudo ocurren en física e ingeniería. Fueron programados en gran parte en FORTRAN y lenguaje ensamblador. Otros sistemas operativos típicos de la época eran FMS (Fortran Monitor System) e IBSYS, el sistema operativo de IBM para el 7094.

Tercera generación (1965-1980). A principios de la década de 1960, la mayoría de los fabricantes de computadoras tenían dos líneas de productos distintas y totalmente incompatibles. Por un lado, estaban los ordenadores científicos a gran escala orientadas a palabras (p.ej. la 7094, que se usaban para cálculos numéricos en ciencia e ingeniería) y por el otro estaban los ordenadores comerciales orientados a caracteres (p.ej. el 1401, que los bancos y las compañías de seguros usaban ampliamente para clasificar e imprimir cintas). Desarrollar y mantener dos líneas de productos completamente diferentes fue una propuesta costosa para los fabricantes.

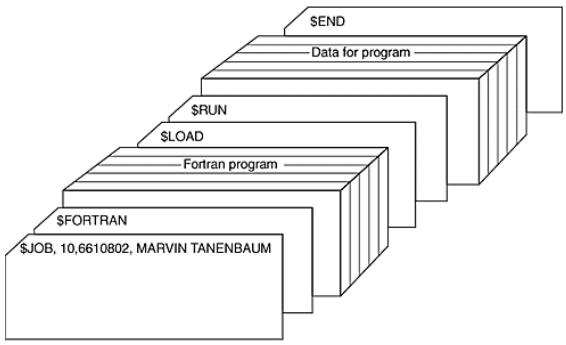

Figure 4: Estructura de un trabajo típico de FMS. Figura extraída de [1].

IBM intentó resolver ambos problemas al presentar el System / 360. El 360 era una serie de máquinas compatibles con software que iban desde el tamaño 1401 hasta mucho más potente que la 7094. De esta forma, una sola familia de máquinas podría satisfacer las necesidades de todos los usuarios. El 360 fue la primera línea de ordenadores importante en utilizar circuitos integrados, lo que proporcionó una gran ventaja de precio / rendimiento sobre las máquinas de segunda generación, que se construyeron a partir de transistores individuales. En los años siguientes, IBM ha presentado sucesores compatibles con la línea 360, utilizando tecnología más moderna, conocida como las series 370, 4300, 3080 y 3090.

A pesar de su enorme tamaño y problemas, OS / 360 y los sistemas operativos similares de tercera generación producidos por otros fabricantes de computadoras satisfacían a la mayoría de sus clientes razonablemente bien. También popularizaron varias técnicas clave ausentes en los sistemas operativos de segunda generación. Probablemente el más importante fue la introducción de la **multiprogramación** fue una parte importante porque permitió que una CPU estuviera ocupada casi el 100% todo el tiempo.

Otra característica importante presente en los sistemas operativos de tercera generación era la capacidad de leer trabajos de tarjetas en el disco tan pronto como se llevaban a la sala de informática. Luego, cada vez que finalizaba un trabajo en ejecución, el sistema operativo podía cargar un nuevo trabajo desde el disco en la partición ahora vacía y ejecutarlo. Esta técnica se llama **spooling** y también se utilizó para la salida. Con el spooling, los 1401 ya no eran necesarios y gran parte del transporte de cintas desapareció.

A pesar de estas mejoras, el tiempo entre el envío de un trabajo y la obtención de la salida era a menudo de varias horas, por lo que una sola coma mal colocada podría provocar que la compilación fallara y que el programador desperdiciara medio día. Esto ayudó a diseñar el concepto de tiempo compartido o **time-sharing**, una variante de la multiprogramación, en la que cada usuario tiene un terminal en línea. El primer sistema de tiempo compartido serio, CTSS (Compatible Time Sharing System), se desarrolló en el MIT en un 7094 especialmente modificado. Sin embargo, el time-sharing no se hizo realmente popular hasta que el hardware de protección necesario se generalizó durante la tercera generación. MIT, Bell Labs y General Electric diseñaron exitosamente una máquina, llamado MULTICS (MULTiplexed Information and Computing Service), que tuvo una gran influencia en los sistemas operativos posteriores. MULTICS permitia la conexión de cientos de usuarios de tiempo compartido de forma simultánea y fue instalada por unas 80 compañías y universidades importantes

en todo el mundo.

Otro desarrollo importante durante la tercera generación fue el fenomenal crecimiento de las **minicomputadoras**, comenzando con el DEC PDP-1 en 1961. Para ciertos tipos de trabajo no numérico, fue casi tan rápido como el 7094 y dio origen a una industria completamente nueva. Fue seguido rápidamente por una serie de otros PDP (a diferencia de la familia de IBM, todos incompatibles) que culminaron en el PDP-11

Uno de los científicos informáticos de Bell Labs que había trabajado en el proyecto MULTICS, Ken Thompson, encontró posteriormente una pequeña minicomputadora PDP-7 que nadie estaba usando y se dispuso a escribir una versión simplificada para un solo usuario de MULTICS. En este trabajo se desarrolló más tarde en el **UNIX** sistema operativo, que se hizo popular en el mundo académico, organismos gubernamentales y muchas empresas.

Figure 5: Foto de MULTICS al MIT. Figura extraída de `https://microteknologias.wordpress.com/2009/11/12/multics-cumple-40-anitos-1969-a-2009/`.

Para hacer posible la compatibilidad de programas en cualquier sistema UNIX, IEEE desarrolló un estándar para UNIX llamado **POSIX**. POSIX define una interfaz mínima de llamada al sistema que deben admitir los sistemas UNIX compatibles. De hecho, algunos otros sistemas operativos ahora también admiten la interfaz POSIX. En 1987, el autor lanzó un pequeño clon de UNIX, llamado **MINIX**, con fines educativos. Funcionalmente, MINIX es muy similar a UNIX, incluido el soporte POSIX.

El deseo de una versión de producción gratuita de MINIX llevó a un estudiante finlandés, Linus Torvalds, a escribir **Linux**. Este sistema se desarrolló en MINIX y originalmente admitía varias funciones de MINIX (por ejemplo, el sistema de archivos MINIX). Desde entonces se ha ampliado a muchos "flavors", pero aún conserva una gran cantidad de estructura subyacente común a MINIX y a UNIX (en el que se basó el primero).

Cuarta generación (1980-presente). Con el desarrollo de circuitos integrados que permiten miles de transistores por centímetro cuadrado, llegó la era de los ordenadores personales. En términos de arquitectura, las ordenadores personales (inicialmente llamadas microcomputadoras) no eran tan diferentes de las minicomputadoras de la clase PDP-11, aunque sí más baratos. Mientras que la

minicomputadora hizo posible que un departamento de una empresa o universidad tuviera su propio ordenador, el chip del microprocesador hizo posible que una sola persona tuviera su propio ordenador personal.

En 1974, Gary Kildall desarrolló un sistema operativo basado en disco llamado **CP/M** (Control Program for Microcomputers) para el Intel 8080 que luego distribuyó a través de una empresa que fundó (Digital Research). Más tarde en 1977, Digital Research reescribió CP/M para hacerlo adecuado para su ejecución en las muchas microcomputadoras que utilizan el 8080, Zilog Z80 y otros chips de CPU. Muchos programas de aplicación se escribieron para ejecutarse en CP/M, lo que le permitió dominar por completo el mundo de la microcomputación durante unos 5 años.

A principios de la década de 1980, IBM diseñó el **PC IBM** y buscó software que se ejecutara en ella. IBM se puso en contacto con **Bill Gates** para obtener la licencia de su intérprete BASIC y para que le proporcionase un sistema operativo. Gates entonces compró el sistema operativo **DOS** (Disk Operating System) a Seattle Computer Products y ofreció a IBM un paquete DOS/BASIC, que IBM aceptó. Gates contrató a la persona que escribió DOS, Tim Paterson, como empleado de la incipiente empresa de Gates, Microsoft, para realizar ciertas modificaciones que pidió IBM. El sistema revisado pasó a llamarse **MS-DOS** (MicroSoft Disk Operating System) y rápidamente llegó a dominar el mercado de PC de IBM. Un factor clave aquí fue la decisión de Gates de vender MS-DOS a empresas informáticas para que lo combinaran con su hardware, en comparación con el intento de Kildall de vender CP/M a los usuarios finales individualmente.

Para cuando IBM PC/AT salió en 1983 con la CPU Intel 80286, MS-DOS estaba firmemente arraigado y CP/M estaba en sus últimas etapas. Más tarde, MS-DOS se utilizó ampliamente en el 80386 y el 80486. Aunque la versión inicial de MS-DOS era bastante primitiva, las versiones posteriores incluyeron características más avanzadas, muchas de ellas tomadas de UNIX. CP/M, MS-DOS y otros sistemas operativos para las primeras microcomputadoras se basaban en que los usuarios escribieran comandos desde el teclado. Esto cambió radicalmente gracias a la investigación de Doug Engelbart en el Instituto de Investigación de Stanford en la década de 1960. Engelbart inventó la **GUI** (interfaz gráfica de usuario), que se completó con ventanas, iconos, menús y ratón. Estas ideas fueron adoptadas por investigadores de Xerox PARC y se incorporaron a las máquinas que construyeron.

Steve Jobs, quien co-inventó la computadora Apple en su garaje, visitó PARC, vio una GUI e instantáneamente se dio cuenta de su valor potencial. Jobs luego se embarcó en la construcción de una Apple con una GUI. Este proyecto llevó al Lisa, que era demasiado caro y fracasó comercialmente. El segundo intento de Jobs, el **Apple Macintosh**, fue un gran éxito, no solo porque era mucho más barato que el Lisa, sino también porque era fácil de usar, lo que significa que estaba destinado a usuarios que no solo no sabían nada de programación, sino que además tenían absolutamente ninguna intención de aprender.

Cuando Microsoft decidió construir un sucesor de MS-DOS, fuertemente influenciado por el éxito de Macintosh, produjo un sistema basado en GUI llamado **Windows**, que originalmente se ejecutaba sobre MS-DOS (era más como un shell que como un verdadero sistema operativo). Durante unos 10 años, de 1985 a 1995, Windows fue solo un entorno gráfico sobre MS-DOS. Sin embargo, a partir

de 1995 se lanzó una versión independiente de Windows, **Windows 95**, que incorporaba muchas características del sistema operativo, utilizando el sistema MS-DOS subyacente solo para arrancar y ejecutar programas antiguos de MS-DOS. En 1998, se lanzó una versión ligeramente modificada de este sistema, llamada **Windows 98**. Sin embargo, tanto Windows 95 como Windows 98 todavía contienen una gran cantidad de lenguaje ensamblador Intel de 16 bits.

Otro sistema operativo de Microsoft es **Windows NT** (NT significa Nueva Tecnología), que es un sistema completo de 32 bits compatible con Windows 95 en un cierto nivel. El diseñador principal de Windows NT fue David Cutler, quien también fue uno de los diseñadores del sistema operativo VAX VMS, por lo que algunas ideas de VMS están presentes en NT. Microsoft esperaba que la primera versión de NT acabaría con MS-DOS y todas las demás versiones de Windows, ya que era un sistema muy superior, pero fracasó. Solo con Windows NT 4.0 finalmente se popularizó a lo grande, especialmente en las redes corporativas. La versión 5 de Windows NT pasó a llamarse **Windows 2000** a principios de 1999. Se pretendía que fuera el sucesor de Windows 98 y Windows NT 4.0. Eso tampoco funcionó del todo, por lo que Microsoft lanzó otra versión de Windows 98 llamada **Windows Me** (Edición Millennium).

En 2001, Microsoft lanzó **Windows XP**, que se construyó en el kernel de Windows NT. Ya en el 2007, se añadieron un gran número de características nuevas de seguridad a partir de un shell rediseñado y nueva interfaz de usuario, y el sistema pasó a llamarse **Windows Vista**. En 2009, apareció el **Windows 7** que introdujo a un gran número de nuevas características con el objetivo de ser compatible con aplicaciones y hardware con los que Windows Vista no era compatible. **Windows 8** lleó en 2012 dónde el Menú Inicio se reemplazó por una pantalla de Inicio de tamaño completo, la cual incluyó nuevas aplicaciones. Su uso está enfatizado para dispositivos con pantallas táctiles, aunque puede ser utilizado con ratón y teclado. Por último, en 2015, Microsoft lanzó **Windows 10** que presenta un conjunto de aplicaciones y una interfaz que permite utilizarse en computadoras personales y dispositivos móviles.

El otro competidor importante en el mundo de las computadoras personales es **UNIX** (y sus diversos derivados). UNIX es más fuerte en estaciones de trabajo y otras computadoras de alta gama, como servidores de red. Es especialmente popular en máquinas impulsadas por chips RISC de alto rendimiento. En las computadoras con Pentium, Linux se está convirtiendo en una alternativa popular a Windows para los estudiantes y cada vez más usuarios corporativos. Aunque muchos usuarios de UNIX, especialmente programadores experimentados, prefieren una interfaz basada en comandos a una GUI, casi todos los sistemas UNIX admiten un sistema de ventanas llamado **X Windows** producido en el MIT. Este sistema maneja la administración básica de ventanas, permitiendo a los usuarios crear, eliminar, mover y cambiar el tamaño de las ventanas usando un ratón. A menudo, una GUI completa, como **Motif**, está disponible para ejecutarse en la parte superior del sistema X Windows, lo que le da a UNIX una apariencia similar a Macintosh o Microsoft Windows, para aquellos usuarios de UNIX que lo deseen.

Un desarrollo interesante que comenzó a tener lugar a mediados de la década de 1980 es el crecimiento de redes de computadoras personales que ejecutan sistemas operativos de red y sistemas operativos distribuidos.

En un **sistema operativo de red**, los usuarios son conscientes de la existencia de varias computadoras y pueden iniciar sesión en máquinas remotas y copiar archivos de una máquina a otra. Cada máquina ejecuta su propio sistema operativo local y tiene su propio usuario (o usuarios) local. Los sistemas operativos de red no son fundamentalmente diferentes de los sistemas operativos de un solo procesador. Obviamente, necesitan un controlador de interfaz de red y algún software de bajo nivel para manejarlo, así como programas para lograr el inicio de sesión remoto y el acceso remoto a los archivos, pero estas adiciones no cambian la estructura esencial del sistema operativo.

Un **sistema operativo distribuido**, por el contrario, es uno que aparece a sus usuarios como un sistema monoprocesador tradicional, aunque en realidad está compuesto por múltiples procesadores. Los usuarios no deben saber dónde se ejecutan sus programas o dónde se encuentran sus archivos; todo eso debería ser manejado de manera automática y eficiente por el sistema operativo. Los verdaderos sistemas operativos distribuidos requieren algo más que agregar un poco de código a un sistema operativo monoprocesador, porque los sistemas distribuidos y centralizados difieren en formas críticas. Los sistemas distribuidos, por ejemplo, a menudo permiten que las aplicaciones se ejecuten en varios procesadores al mismo tiempo, por lo que requieren algoritmos de programación de procesadores más complejos para optimizar la cantidad de paralelismo. Los retrasos en la comunicación dentro de la red a menudo significan que estos (y otros) algoritmos deben ejecutarse con información incompleta, desactualizada o incluso incorrecta. Esta situación es radicalmente diferente de un sistema de un solo procesador en el que el sistema operativo tiene información completa sobre el estado del sistema.

1.3 Taxonomía de los sistemas operativos

Como se comentó anteriormente, un sistema operativo es el software básico de las máquinas (ordenador, móvil, televisión, etc.) que proporciona una interfaz entre los programas informáticos y el hardware. Las funciones básicas del sistema operativo son administrar los recursos de la máquina (tiempo de CPU, memoria, etc.), coordinar el hardware y organizar archivos y directorios en dispositivos de almacenamiento.

Durante la breve historia descrita en el apartado anterior, se ha mencionado la evolución de los sistemas operativos. Cada generación de ordenadores han ido incluyendo diversas mejoras en la tecnología (tarjetas perforadas, transistores, microchips) que han también ayudado a incorporar mejoras en los sistemas operativos. De acuerdo al libro de "Modern Operating System" de Tanenbaum [1], los sistemas operativos los podríamos clasificar en:

- **Mainframe**. Ordenador central que pueden ocupar el tamaño de una habitación y que se utilizan en grandes compañias, centros de datos o incluso para almacenar servidores web. Este tipo de sistemas tiene una gran capacidad para manejar información de Entrada/Salida y están orientados a procesar muchos procesos simultáneamente. Algunos de estos sistemas operativos permiten cientos o incluso miles de usuarios simultáneamente.
- **Servidores**. Corresponde a un nivel por debajo de los Mainframe. Se ejecutan en servidores y también permiten múltiples usuarios conectarse al mismo tiempo, que comparten recursos de software y hardware.

- **Multiprocesadores**: Estos sistemas operativos son una variación de los servidores. Admiten el proceso del de un mismo programa en más de una CPU, y dependiendo de como estan conectados y que recursos comparten se pueden llamar "computadora paralela", "multicomputadora" o "multiprocesador".
- **Ordenador personal**: En la mayoria de estos sistemas operativos, éste está optimizado para ofrecer una buena interface a un usuario único. Manejan principalmente editores de texto, hojas de cálculo y navegación por internet.
- **Tiempo real**: Estos sistemas operativos responden a la entrada al instante. Sobre todo en la industria, la mayoria de los sistemas operativos que controlan maquinaria pecisan de respuesta instanánea. Los sistemas operativo DOS y UNIX, no funcionan en tiempo real.
- **Empotrados** Aquí nos encontramos con los sistemas operativos de dispositivos pequeños como móviles, libros electrónicos u otros dispositivos pequeños. Suelen tener características de sistemas operativos en tiempo real, pero por norma general disponen de recursos limitados (memoria, potencia, etc).
- **Smart Cards**. Contienen secuencias de instrucciones incrustadas permanentemente en la ROM de la tarjeta inteligente. Se utilizan con frecuencia en procesos y aplicaciones en tarjeta. Proporcionan funciones para el intercambio de datos y comandos, almacenamiento de datos, procesamiento de datos, procesos criptográficos, etc.

De acuerdo a sus funciones, podemos clasificar los sistemas operativos de acuerdo a sus funciones:

- **Gestión de procesos**: El procesador del sistema operativo gestiona la distribución de los diferentes programas compartiendo el uso de la CPU.
- **Gestión de memoria**: El sistema operativo administra el espacio de memoria asignado para cada aplicación y cada usuario. Cuando la memoria física es insuficiente, el sistema operativo crea un área de memoria en el disco duro, llamada "memoria virtual". A pesar de ser más lenta, la memoria virtual le permite ejecutar aplicaciones que requieren una capacidad de memoria superior a la RAM disponible en el sistema.
- **Gestión de entrada / salida**: El sistema operativo unifica y controla el acceso a los programas de recursos hardware a través de los controladores (administradores periféricos o de entrada/salida).
- **Gestión de aplicaciones**: El sistema operativo debe garantizar que las aplicaciones se ejecuten sin problemas al asignar los recursos que necesitan para funcionar. Esto significa que si una aplicación no responde correctamente, ésta puede ser parada por el sistema operativo.
- **Gestión de seguridad**: El sistema operativo es responsable de la seguridad en relación con la ejecución de programas y debe garantizar que los recursos se utilizan solo para programas y usuarios con la autorización adecuada.
- **Gestión de archivos**: El sistema operativo administra toda la escritura y lectura en el sistema de archivos y los permisos de acceso a archivos y aplicaciones de usuario.

- **Gestión de la información**: El sistema operativo debe proporcionar cientos de indicadores que se pueden utilizar para diagnosticar el funcionamiento del equipo en un momento dado.

Por otro lado, Anderson y Dahli [2] describen 3 roles principales que juegan los sistemas operativos:

- **Árbitro**. El sistema operativo gestiona los recursos de una máquina real entre las múltiples aplicaciones que se ejecutan. El sistema operativo debe aislar las aplicaciones entre sí para evitar que un fallo o malintención en una aplicación afecte a las otras aplicaciones así como el mismo sistema operativo, al mismo tiempo que debe permitir la comunicación entre aplicaciones.
- **Ilusionista**. El sistema opertivo hace de ilusionistas también. Aunque las aplicaciones comparten recursos físicos de la máquina, el sistema operativo les hace creer que tienen todos los recursos para ellos (infinita memoria, 100% de la CPU, ...)
- **Pegamento**. Muchos sistemas operativos proveen de rutinas de interfaz común a las aplicaciones (look and feel). También ofrece una capa de separación entre hardware y software, que permite a las aplicaciones ser escritas independiente del dispositivo en el que se ejecuten.

1.4 Conclusiones

Al igual que la tecnologia hardware detrás de los ordenadores, los sistemas operativos han evolucionando mucho desde la aparición del primer sistema operativo (GMOS) para los IBM 701 en el año 1955. Con el tiempo, se han ido incorporando características como el multiproceso de tareas, la virtualización de los recursos, la conexión multiusuario; siempre buscando un entorno fiable, eficiente y seguro. En este capítulo se ha situado el contexto histórico de los sistemas operativos, concluyendo que existe un gran número de sistemas operativos en la actualidad. Cada uno con sus propias características y capacidades, y en constante evolución de acuerdo a las necesidades de los usuarios y la tecnología existente en cada momento.

Bibliography

[1] Andrew S Tanenbaum and Herbert Bos. *Modern operating systems*. Pearson, 2015.

[2] Thomas Anderson and Michael Dahlin. *Operating Systems: Principles and Practice*, volume 2. Recursive books, 2014.