SISTEMAS INFORMÁTICOS UD-1

CONCEPTOS BÁSICOS DE LOS SISTEMAS INFORMÁTICOS (II)

EL SISTEMA OPERATIVO

ÍNDICE

[INTRODUCCIÓN 3](#_Toc117605189)

[FUNCIONES Y ESTRUCTURA 3](#_Toc117605190)

[EVOLUCIÓN 5](#_Toc117605191)

[CLASIFICACIÓN 7](#_Toc117605192)

[LICENCIAS 8](#_Toc117605193)

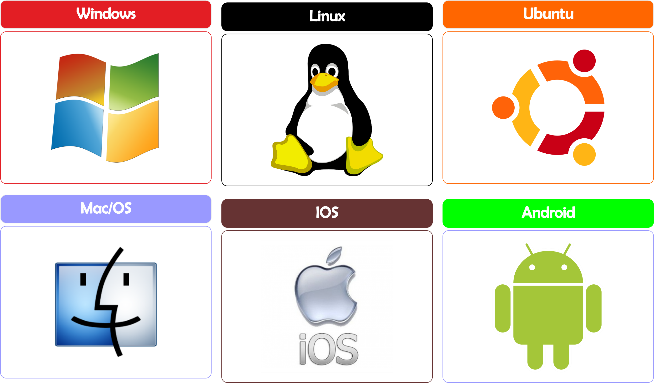
[A. COMERCIALES, PROPIETARIAS O PRIVATIVAS 8](#_Toc117605194)

[B. GRATUITAS, LIBRES O ABIERTAS: 9](#_Toc117605195)

[C. DIFERENCIA ENTRE DERECHOS DE AUTOR, COPYLEFT Y COPYRIGHT 10](#_Toc117605196)

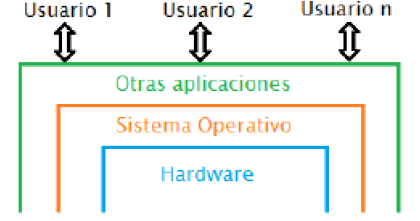
[CREATIVE COMMONS 11](#_Toc117605197)

# INTRODUCCIÓN

Es el **primer programa** **que se ejecuta** cuando iniciamos el ordenador y el **último que cesa su funcionamiento** cuando lo apagamos.

Los programas se ejecutan sobre el sistema operativo y hacen uso de los diferentes hardware que necesitan, pero no directamente sino a través del sistema operativo.

Por tanto, el SO es una **interfaz** (un intermediario) entre el hardware y el software.

Aporta **transparencia** (abstrae a aplicaciones y usuarios de las características del hardware) y **portabilidad** (un software se puede ejecutar en diferentes dispositivos).

Esto permite que los elementos de ambas capas (software y hardware -e incluso los usuarios-) se abstraigan de los detalles de la comunicación, es decir, el hardware ejecuta las aplicaciones sin conocer sus características y las aplicaciones actúan sobre el hardware sin conocer su arquitectura.

Los **objetivos** del SO son, por tanto:

* + **Abstracción**. El hardware y el software son independientes en su ejecución sin tener en cuenta las características o arquitectura del otro.
  + **Seguridad**. Impide que haya intrusiones de unos procesos en las funciones de memoria de otros procesos.
  + **Gestión de recursos**. Todos los recursos hardware que se encuentran dentro del ordenador son gestionados por el SO.

# FUNCIONES Y ESTRUCTURA

**Diagrama

Descripción generada automáticamente**Las principales funciones de del SO son:

* **Control de ejecución de programas**. También conocida como la gestión de los procesos. Desde que se ejecuta un proceso, este debe tomar el control de la CPU y el SO le tiene que dar permiso para ello. Igualmente, el SO es el responsable de que, una vez terminado el proceso, la CPU quede libre. También controla el orden en el que se ejecutan los procesos y cuales tienen prioridad por lo que el orden de ejecución de los procesos no tiene por qué ser por orden de llegada a la cola.
* **Administración de los periféricos**. A través de los drivers, el SO es capaz la administración de los diferentes periféricos que se encuentren conectados al sistema informático.
* **Gestión de permisos y de usuarios**. Diferentes usuarios pueden tener diferentes permisos/roles para el uso de los recursos del sistema (administrador, usuarios estándar, etc.), el SO se encarga de decidir qué puede hacer cada usuario (sólo lectura, procesos reservados a administradores, etc.).
* **Control de concurrencia**. La concurrencia ocurre cuando se ejecutan varios procesos al mismo tiempo (esto siempre ocurre). El SO se encarga de gestionar qué proceso toma el control de la CPU.
* **Control de errores**. Durante la ejecución del SO se pueden dar varios errores debido a todo el software que se está ejecutando sobre él. El SO debe detectar y corregir los errores que surjan en tiempo de ejecución, además, debe evitar que haya una pérdida de datos por error (por ejemplo, el pantallazo azul sería un ejemplo de un mal control de errores y un buen ejemplo sería la ralentización del dispositivo).
* **Administración de memoria**. El SO debe tener control de la memoria determinando qué partes de la memoria están liberadas, qué partes están ocupadas y, en el caso de que la memoria esté ocupada, comprobar si es necesaria la creación de una memoria virtual.
* **Diagrama

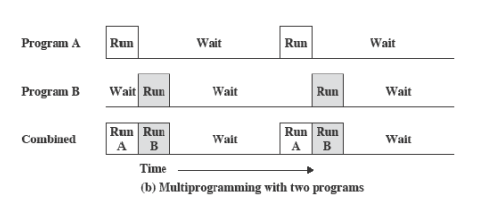
  Descripción generada automáticamenteControl de seguridad**. El SO impide que unos procesos puedan crear intrusiones en las funciones de memoria de otros procesos (esto es lo que realizan los virus).

La estructura del SO es:

* **Kernel**. Es el núcleo del SO. Se carga en la RAM cuando se inicia el sistema y se encarga de gestionar el hardware cuando iniciamos el SO.
* **Capa de servicios**. Permite a las aplicaciones hacer uso del hardware, es decir, proporciona una serie de servicio a las aplicaciones que se ejecutan sobre el SO y las aplicaciones ya saben los servicios que van a poder utilizar.
* **Shell**. Es el programa informático que interpreta las órdenes del usuario (bien por consola de comandos o no).

# EVOLUCIÓN

* **Proceso en serie (Fin. 40s – Med. 50s)**. **No existía el SO** y el programador interactuaba directamente con el hardware a través de una **consola** con indicadores luminosos, un dispositivo de entrada, conmutadores y una impresora. Los programas eran en **código máquina** y se cargaban a través de dispositivos de entrada (tarjetas). Hasta que no terminaba una tarjeta no se podía usar otra.
* **Proceso sencillo por lotes (Med. 50s)**. Parte de una idea software llamada **monitor**. Los usuarios ya no accedían directamente a la máquina. Los trabajos se agrupaban por lotes en el dispositivo de entrada y el monitor decidía la secuencia a ejecutar de forma automática. Una vez que terminaba el procesamiento del programa este debía volver al monitor (**Lenguaje de Control de Trabajos o JCL**). Una gran parte del monitor debía residir siempre en la memoria principal, es lo que se conocía como **monitor residente**.
* **Multiprogramación o multitarea (1960)**. Consiste en ejecutar múltiples programas al mismo tiempo para con ello aprovechar el tiempo de inactividad del procesador que se produce durante las operaciones de entrada o salida de los programas que se están ejecutando. Para que estos SOs funcionen se requiere que la memoria sea suficiente como para ejecutar al menos 2 programas al mismo tiempo.



El programa A comienza su ejecución (tomando el control de la CPU) y cuando tiene que realizar una operación de entrada/salida, el SO da el control de la CPU al programa B que comienza su ejecución, una vez que este programa B tiene que hacer una operación de entrada/salida, el SO le da el control al programa A y así de forma sucesiva.

Siguen existiendo tiempos de inactividad, porque los procesos de entrada/salida son más lentos que otro tipo de procesos, pero son tiempos más cortos en comparación a una ejecución de los programas de 1 en 1 por lo que se consigue optimizar el uso del procesador.

* **Tiempo compartido (1970)**. Parte de la idea de sistemas hardware compartidos entre varios usuarios (los cuales usan terminales para comunicarse de manera simultánea con el ordenador) en situaciones en las que cada usuario tiene al menos un programa en memoria. El SO se encarga de dar la apariencia de estar ejecutando múltiples procesos al mismo tiempo y ofrecer, de esa manera, una pequeña porción del tiempo del procesador a cada usuario.

Fusiona el concepto de los SO multiprogramación de ejecutar varios procesos al mismo tiempo para aprovechar los tiempos de espera y el propio concepto del tiempo compartido de minimizar el tiempo de respuesta.

* **Sistemas operativos en red**. Se trata de conectar 2 o más ordenadores a través de algún medio de comunicación (físico o no) con el objetivo de poder compartir los diferentes recursos software. No es igual que el tiempo compartido porque en este caso no se comparte el hardware, sino que cada usuario tiene el suyo propio. Tiene un controlador de red y programas que permiten acceder a la red.
* **Sistemas operativos distribuidos**. Se trata de un SO común, compartido por una red de ordenadores, que proporciona a los usuarios la posibilidad de usar los recursos disponibles en todos los ordenadores que la forman (todos los equipos son independientes, pero actúan como un único equipo). El acceso a esos recursos es transparente, el usuario no sabe exactamente el hardware que está utilizando.
* **Sistemas operativos en tiempo real**. En estos SOs hay un control total sobre los tiempos de las aplicaciones y procesos se sabe cuándo comienzan y cuándo terminan y el SO debe responder siempre dentro de unos márgenes de tiempo estrictos requeridos por el sistema controlado para no fallar.

# CLASIFICACIÓN

|  |  |  |  |
| --- | --- | --- | --- |
| **SEGÚN** | **TIPOS** | **DEFINICIÓN** | **EJEMPLOS** |
| **Número de usuarios** | **Monousuario** | No distinguen entre diferentes perfiles de usuarios. | Windows 2000  MS-DOS |
| **Multiusuario** | Distingue entre diferentes perfiles de usuarios. | Windows NT  Linux |
| **Número de procesos concurrentes (al mismo tiempo)** | **Monoproceso** | Sólo son capaces de manejar un proceso por vez. | MS-DOS  Windows ‘95 |
| **Multiproceso** | Son capaces de gestionar y realizar varios procesos al mismo tiempo. | Windows NT 4.0  Windows 2000 Professional |
| **Número de procesadores soportados** | **Monoprocesador** | Son aquellos que sólo son capaces de manejar un único procesador por lo que si el ordenador tuviese más de uno sería inútil. | MS-DOS  Mac OS |
| **Multiprocesador** | Son aquellos que pueden utilizar varios procesadores en un mismo sistema informático siendo capaz de utilizarlos de forma simultánea para distribuir la carga de trabajo. | UNIX  Linux |
| **Tiempo de respuesta** | **Tiempo compartido** | Permiten que un único procesador realice más de un proceso en el mismo tiempo. Simulan simultaneidad. | UNIX  Multics |
| **Tiempo real** | Garantizan el tiempo de respuesta de un proceso evitando simultanear varios procesos en un solo procesador. | Sistemas militares |
| **Manejo de recursos** | **Centralizado** | Utiliza los recursos hardware de un único ordenador. | SO domésticos |
| **En red** | Posibilita la conexión y comunicación de varios ordenadores dentro de una red. Tiene la capacidad de interactuar con los SOs de los ordenadores conectados y dar respuesta. | Mac OS X  Windows Server |
| **Distribuido** | Permite utilizar los recursos de más de un ordenador al mismo tiempo de forma transparente para el usuario. | LOCUS  Solaris-MC |

También existe una clasificación en función de la forma en la que estén disponibles.

|  |  |  |  |
| --- | --- | --- | --- |
| **SEGÚN** | **TIPOS** | **DEFINICIÓN** | **EJEMPLOS** |
| **Su coste económico** | ***Freeware*** | Son gratuitos. | Ubuntu  Linux |
| **Comerciales** | Son de pago. | Windows  OSX |
| **Su propiedad intelectual** | **Libres\*** | Se pueden usar, distribuir, acceder y modificar libremente. | Ubuntu |
| **Propietarias** | Son SO cuya propiedad es de la empresa que los ha desarrollado. La empresa no vende el SO sino una licencia. No se tiene acceso al código fuente o, por lo menos, no se tiene permiso para modificarlo libremente. | Windows |
| **Su acceso al código fuente** | ***Open Source*** | Permiten el acceso libre al código. | Ubuntu |
| **Cerradas (privativas)** | No permiten el acceso al código. | Windows |

\* Para que un software sea considerado libre debe garantizar 4 cosas:

* **Libertad 0: Libertad de uso.** Libertad para usar el programa con cualquier propósito.
* **Libertad 1: Libertad de estudio.** Libertad para estudiar cómo funciona el programa y modificarlo, adaptándolo a las propias necesidades.
* **Libertad 2: Libertad de distribución.** Libertad de distribuir copias del programa.
* **Libertad 3: Libertad de mejora.** Libertad de mejorar el programa y hacer públicas esas mejoras a los demás de forma que toda la comunidad se beneficie.

# LICENCIAS

Son los contratos o acuerdos legales que se establecen entre los usuarios del software y su autor.

Su objetivo es describir las obligaciones y derechos de los usuarios y autores del software.

## COMERCIALES, PROPIETARIAS O PRIVATIVAS

* + - **OEM**. Cuando se compra un producto (ordenador) este viene con un sistema operativo asociado, tiene licencia de uso en ese único sistema.
    - **Retail**. Se compra directamente al desarrollador. Se es propietario de la licencia, podemos instalarlo en cualquier tipo de hardware compatible.
    - **VLM (por volumen)**. Se compran por bloques y son utilizadas por empresas. Con una única clave de licencia se puede instalar el SO en varios equipos a la vez.
    - **MSDN**. Sólo se puede usar para actividades educativas y de formación. Si se usa en equipos fuera de estas actividades, es ilegal.

## GRATUITAS, LIBRES O ABIERTAS:

* + - **Licencia Pública General (GPL o GNU):**
      * De las más utilizadas.
      * Tipo especial de licencia ***copyleft***: Se puede usar, modificar y distribuir el programa bajo el mismo tipo de licencia.
      * El software debe ser gratuito y asegurarse siempre de mantener los nombres y créditos de los autores originales.
    - **Licencia Pública General Reducida (LGPL):**
      * Más permisiva que la GPL.
      * Permite enlazar programas con esta licencia a módulos de software que no sean libres.
    - **Licencia pública general Affero (AGPL):** 
      * Mismas condiciones que la GPL.
      * Obligación añadida de distribuir el software si éste sirve para ofrecer servicios a través de una red de ordenadores.
    - **Licencia de distribución de software Berkeley (BSD):**
      * Utilizada en gran cantidad de software distribuido junto a los sistemas operativos BSD.
      * No establece ninguna restricción en lo referente a la utilización del código para obtener otros productos.
      * Permite utilizar, modificar y distribuir a otras personas el código fuente y el código binario con o sin modificaciones, pudiendo licenciarse con el tipo de licencia que se desee.
      * No es obligatorio mencionar a los autores ni proporcionar el código fuente.
    - ***Mozilla Public License* (MPL)*:***
      * Permite una reutilización comercial y no libre del software, sin restringir la reutilización del código ni el relanzamiento bajo la misma licencia.
      * Aunque se creó para el navegador Mozilla es utilizado por los desarrolladores para la liberación de código fuente.
    - **Licencia MIT:**
      * Creado en el Instituto tecnológico de Massachusetts.
      * Permite la creación de software libre o no libre modificando productos con esta licencia.
      * Permite no liberar los cambios realizados al programa original.

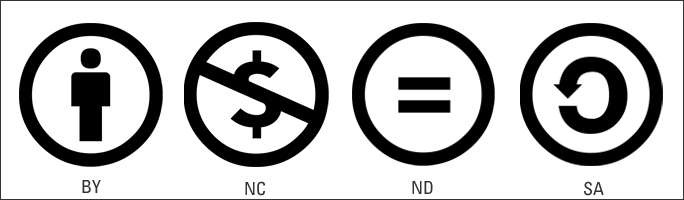
## DIFERENCIA ENTRE DERECHOS DE AUTOR, *COPYLEFT* Y *COPYRIGHT*

* + - **Derechos de autor**: Conjunto de normas y principios que afirman los derechos morales y patrimoniales que la ley concede a los autores por el simple hecho de la creación de una obra, esté publicada o inédita. Cubre tanto al autor como a la obra.
    - ***Copyleft***: Es una práctica legal que consiste en el ejercicio del derecho de autor con el objetivo de propiciar el libre uso y distribución de una obra, exigiendo que los concesionarios preserven las mismas libertades al distribuir sus copias y derivados.
    - ***Copyright***: Tipo de propiedad intelectual que le otorga a su dueño el derecho exclusivo de copiar, distribuir, adaptar, exhibir y producir obras creativas, por lo general, por un tiempo limitado. Su objetivo es proteger la expresión original de una idea asociada a una obra creativa, no la propia idea. Está sujeto a limitaciones y excepciones basadas en consideraciones de interés público.

# CREATIVE COMMONS

Licencia para ayudar a los creadores a mantener sus derechos de autor al mismo tiempo que permiten a otras copiar, distribuir y hacer algunos usos de su obra. Existen varias combinaciones en función de 4 características:

* **Atribución (BY)**: Conocer quién ha creado la obra original, siempre se debe mantener para preservar al autor original. Aparece en todas las combinaciones.
* **No comercial (NC)**: Se puede copiar, distribuir y representar o hacer obras derivadas, pero no se puede comercializar.
* **No derivadas (ND)**: Se puede copiar, distribuir, exhibir y representar, pero no se puede modificar.
* **Compartir igual (SA)**: Se debe distribuir obras modificadas bajo la licencia idéntica a la original.



Estas 4 características dan lugar a 6 combinaciones: