

BLOQUE II

TEMA 04

Sistemas operativos. Características y elementos constitutivos. Sistemas Windows. Sistemas Unix y Linux. Sistemas operativos para dispositivos móviles.

1. SISTEMA OPERATIVOS: DEFINICION, FUNCIONES, EVOLUCION Y CLASIFICACIÓN

1.1 DEFINICIÓN Y FUNCIONES

Un sistema operativo (SO) es un conjunto de programas informáticos que se ejecutan en modo privilegiado y que administran los recursos físicos y lógicos de un dispositivo, como la memoria, el procesador, los dispositivos de entrada/salida y el almacenamiento.

Actúa como **intermediario entre el hardware y el software** de aplicación, permitiendo que los programas funcionen sin tener que gestionar directamente los componentes físicos. Se inicia automáticamente al encender el equipo y permanece activo durante toda la sesión de uso,

Gráficamente, la relación entre las diferentes capas Software existentes entre el Hardware de la máquina y las aplicaciones finales se pueden representar de la siguiente manera:



El SO actúa como una capa intermedia entre el hardware y los programas de aplicaciones. Se encarga de interpretar las órdenes y adecuarlas al hardware para que el usuario trabaje más fácilmente. Debe realizar diferentes funciones como son:

Gestión de procesos \equiv Controla la ejecución de los procesos (programas en ejecución). Asigna tiempo de CPU a cada proceso y gestiona su estado (listo, ejecutando, bloqueado). Coordina la multitarea y la concurrencia.

Gestión de la memoria \equiv Administra la memoria RAM disponible. Asigna y libera espacio de memoria a los procesos según lo necesiten. Implementa técnicas como la memoria virtual para optimizar el uso.

Gestión del sistema de archivos \equiv Organiza y controla el acceso a los archivos almacenados en discos y otros medios. Proporciona estructuras jerárquicas (carpetas, directorios). Controla permisos de lectura, escritura y ejecución.

BLOQUE II

TEMA 04

Gestión de dispositivos de entrada/salida \equiv Coordina el uso de periféricos como teclado, ratón, impresoras, discos, etc. Usa controladores (drivers) para comunicarse con el hardware. Implementa colas y prioridades para el acceso a dispositivos.

Interfaz con el usuario \equiv Proporciona una interfaz gráfica (GUI) o de línea de comandos (CLI). Permite al usuario interactuar con el sistema y ejecutar programas.

Gestión de la seguridad y control de acceso \equiv Protege los recursos del sistema frente a accesos no autorizados. Administra cuentas de usuario, contraseñas y permisos. Aísla procesos para evitar interferencias o ataques.

Comunicación entre procesos (IPC) \equiv Facilita el intercambio de datos entre procesos que se ejecutan simultáneamente. Usa mecanismos como pipes, colas de mensajes o memoria compartida.

Modos de operación o ejecución

Todos los sistemas operativos proveen dos modos de ejecución: modo usuario y modo kernel. Estos modos determinan el nivel de acceso que tiene un proceso al hardware y a los recursos críticos del sistema.

Un bit, denominado bit de modo, se añade al hardware de la computadora para indicar el modo actual kernel(0) o usuario(1).

Las aplicaciones no deben poder usar todas las instrucciones de la CPU. No obstante, el Sistema Operativo, tiene que poder utilizar todo el conjunto de instrucciones del CPU. Por ello, una CPU debe tener (al menos) dos modos de operación diferentes.

- **Modo Kernel o supervisor** \equiv Es un modo de ejecución del sistema operativo en el cual se tiene un modo de funcionamiento privilegiado. En este modo **todas las instrucciones de hardware están disponibles**. El sistema operativo es el único que debe ejecutarse en este modo.
 - Tiene acceso completo al hardware y a todos los recursos del sistema.
 - Puede ejecutar cualquier instrucción de la CPU.
 - Es responsable de la gestión de memoria, procesos, dispositivos y seguridad
 - Ejemplos: Planificador de procesos, controladores de dispositivos, gestor de memoria
- **Modo usuario** \equiv En este modo de ejecución se puede ejecutar un **conjunto reducido de instrucciones** de hardware. Los procesos a nivel de usuarios se ejecutan en este modo.
 - No puede acceder directamente al hardware ni a áreas críticas del sistema.
 - Si necesita servicios del sistema operativo, debe hacer una llamada al sistema (system call).
 - Aísla errores: si una aplicación falla, no compromete todo el sistema.
 - Ejemplos: Navegadores, juegos, etc.

Los sistemas operativos están controlados mediante **interrupciones**. Si no hay ningún proceso que ejecutar, ningún dispositivo de E/S al que dar servicio y ningún usuario al que responder, el sistema operativo debe permanecer inactivo, esperando a que algo ocurra. Los sucesos casi siempre se indican mediante la ocurrencia de una interrupción o **excepción**. Una **excepción** es

BLOQUE II

TEMA 04

una interrupción generada por software, debida a un error o a una solicitud específica de un programa de usuario. Genéricamente se pueden definir como:

- **Interrupción** \equiv señal que envía un dispositivo de E/S a la CPU para indicar que la operación de la que se estaba ocupando, ya ha terminado.
- **Excepción** \equiv una situación de error detectada por la CPU mientras ejecutaba una instrucción, que requiere tratamiento por parte del SO.

1.1 EVOLUCION

PRIMERA GENERACIÓN

Proceso por lotes (batch processing) \equiv A principios de los años 1950 con el objetivo de facilitar la interacción entre persona y computadora, los sistemas operativos hacen una aparición discreta y bastante simple, con conceptos tales como el monitor residente y el almacenamiento temporal.

- Uso de tarjetas perforadas.
- Sistema monousuario.
- Introducción de monitores residentes para automatizar la carga de programas.

SEGUNDA GENERACIÓN

Multiprogramación \equiv Varios programas cargados en memoria simultáneamente. Mejora en la utilización del procesador.

La programación multitareas o los sistemas multiprogramados buscaban maximizar el tiempo de uso efectivo del procesador ejecutando varios procesos al mismo tiempo.

Se introducen sistemas como:

- Sistemas de tiempo compartido
- Sistemas en tiempo real
- Sistemas multiprocesador.
 - *Arquitectura NUMA* (Non-Uniform Memory Access)
 - *Arquitectura SMP* (Symmetric Multi-Processing).

TERCERA GENERACIÓN

Sistema multiusuarios.

- Soporte para múltiples usuarios conectados a terminales.
- Sistemas operativos más complejos con gestión de memoria virtual.
- Aparición de UNIX (1969), base de muchos sistemas modernos.

BLOQUE II

TEMA 04

CUARTA GENERACIÓN

- Popularización de computadoras personales.
- Interfaces gráficas de usuario (GUI).
- Sistemas como MS-DOS, Windows, Mac OS.

Se desarrollan las microcomputadoras, o sea, computadoras personales o PC. Se desarrollan las supercomputadoras.

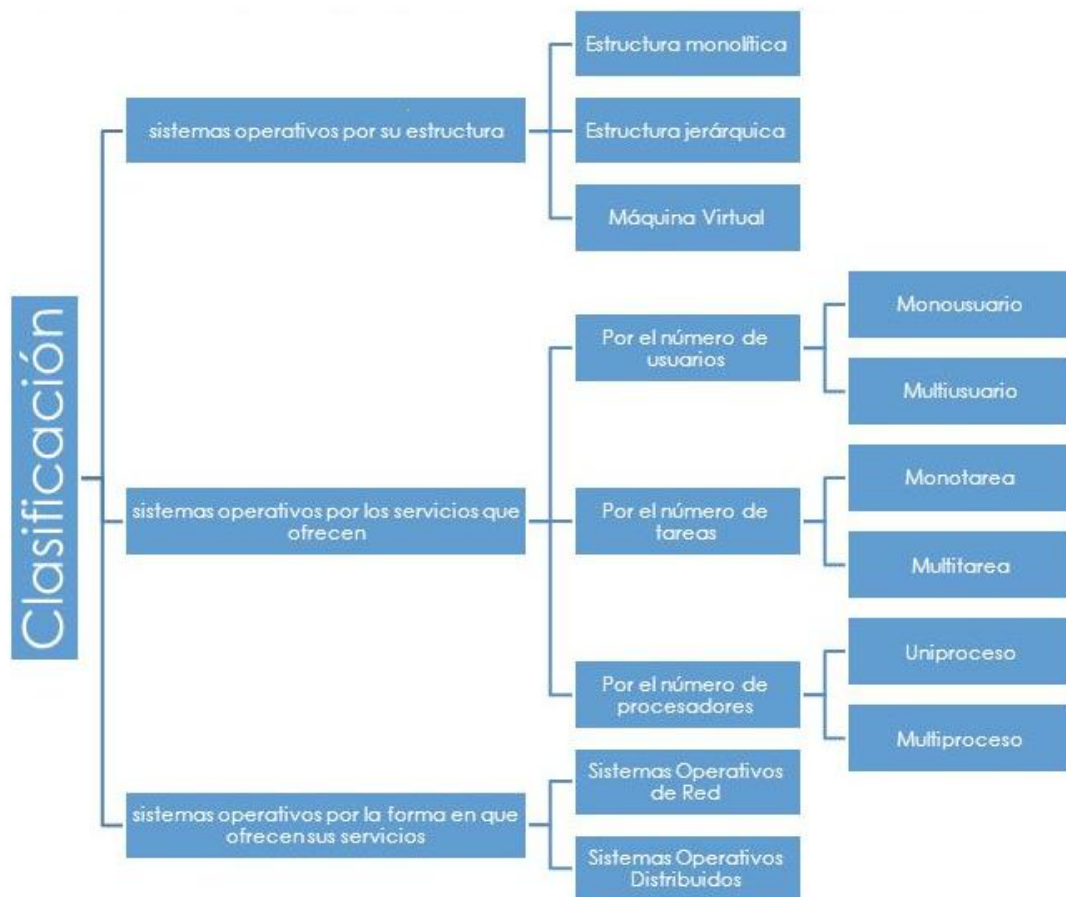
QUINTA GENERACIÓN

- Sistemas operativos para móviles (Android, iOS).
- Integración con redes, nube y seguridad avanzada.
- Interfaces táctiles, asistentes virtuales, y soporte para inteligencia artificial

Hoy en día, los sistemas operativos se adaptan a entornos híbridos (local + nube), soportan contenedores (como Docker), y se integran con tecnologías emergentes como IA, realidad aumentada y computación cuántica.

1.2 CLASIFICACIÓN

Tipos de Sistemas Operativos



Existen diversas clasificaciones de SO, atendiendo a diversos criterios:

A. POR SU ESTRUCTURA

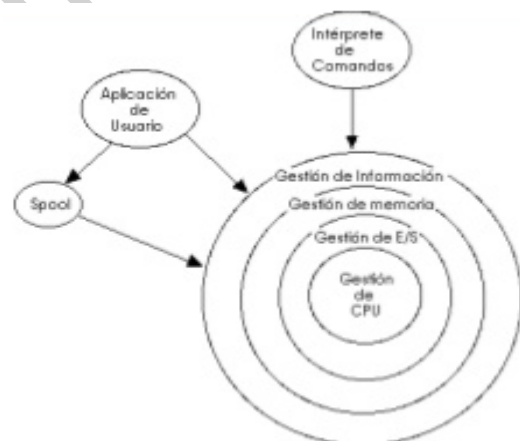
- ❖ **Estructura Monolítica:** Es la estructura de los primeros sistemas operativos constituidos fundamentalmente por un solo programa compuesto de un conjunto de rutinas entrelazadas de tal forma que cada una puede llamar a cualquier otra. Las características fundamentales de este tipo de estructura son:

BLOQUE II

TEMA 04

- Construcción del programa final a base de módulos compilados separadamente que se unen a través del enlazador. Buena definición de parámetros de enlace entre las distintas rutinas existentes, que puede provocar mucho acoplamiento.
 - Carecen de protecciones y privilegios al entrar a rutinas que manejan diferentes aspectos de los recursos de la computadora, como memoria, disco, etc.
 - Generalmente están hechos a medida, por lo que son eficientes y rápidos en su ejecución y gestión, pero por lo mismo carecen de flexibilidad para soportar diferentes ambientes de trabajo o tipos de aplicaciones.
- ❖ **Estructura Jerárquica:** A medida que fueron creciendo las necesidades de los usuarios y se perfeccionaron los sistemas, se hizo necesaria una mayor organización del software, del sistema operativo, donde una parte del sistema contenía sub-partes y esto organizado en forma de niveles.

Se dividió el sistema operativo en pequeñas partes, de tal forma que cada una de ellas estuviera perfectamente definida y con un claro interface con el resto de elementos.



En la estructura anterior se basan prácticamente la mayoría de los sistemas operativos actuales. Otra forma de ver este tipo de sistema es la denominada de anillos concéntricos o "rings". En el sistema de anillos, cada uno tiene una apertura, conocida como puerta o trampa (trap), por donde pueden entrar las llamadas de las capas inferiores. De esta forma, las zonas más internas del sistema operativo o núcleo del sistema estarán más protegidas de accesos indeseados desde las capas más externas. Las capas más internas serán, por tanto, más privilegiadas que las externas.

- ❖ **Máquina Virtual:** Se trata de un tipo de sistemas operativos que presentan una interface a cada proceso, mostrando una máquina que parece idéntica a la máquina real subyacente. Estos sistemas operativos separan dos conceptos que suelen estar unidos en el resto de sistemas: la multiprogramación y la máquina extendida. El objetivo de los sistemas operativos de máquina virtual es el de integrar distintos sistemas operativos dando la sensación de ser varias máquinas diferentes.

BLOQUE II

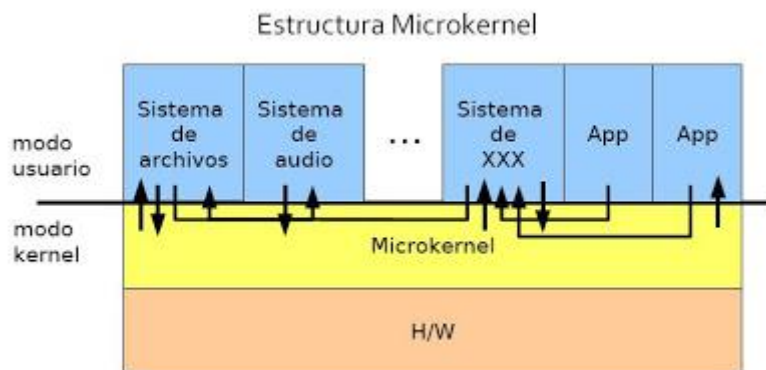
TEMA 04

El núcleo de estos sistemas operativos se denomina monitor virtual y tiene como misión llevar a cabo la multiprogramación, presentando a los niveles superiores tantas máquinas virtuales como se soliciten. Estas máquinas virtuales no son máquinas extendidas, sino una réplica de la máquina real, de manera que en cada una de ellas se pueda ejecutar un sistema operativo diferente, que será el que ofrezca la máquina extendida al usuario

- ❖ **Cliente-Servidor (Microkernel):** El tipo más reciente de sistemas operativos es el denominado Cliente-servidor, que puede ser ejecutado en la mayoría de las computadoras, ya sean grandes o pequeñas. Este sistema sirve para toda clase de aplicaciones; por tanto, es de propósito general y cumple con las mismas actividades que los sistemas operativos convencionales.

El núcleo o kernel es la parte fundamental de un sistema operativo ya que es la responsable de facilitar el acceso seguro al hardware de la forma más básica. Toda esta comunicación lo hace a través del Shell, que es un intérprete de comandos.

Entonces al referirnos a una estructura basada en Microkernel, nos referimos a la estructura de Sistema Operativo, que está basada en un tipo de kernel que provee un conjunto de llamadas primitivas al sistema, para poder implementar servicios básicos, como por ejemplo la comunicación entre procesos y la gestión del espacio de direcciones.



BLOQUE II

TEMA 04

Entre las principales ventajas que nos brinda este tipo de estructura de Sistema Operativo podemos mencionar las siguientes:

- Reduce la complejidad (Simplicidad)
- Descentralización de los fallos
- Modularidad
- Facilita la extensión del Sistema Operativo

Y entre las desventajas están las siguientes:

- Bajo desempeño debido a las llamadas primitivas al sistema
- Complejidad en la sincronización modular

Sistemas Operativos que tenían este tipo de estructura de Microkernel: Minix, Hurd, L4, Amoeba, QNX, RadiOS

- ❖ **Híbrido:** «Híbrido» implica que el núcleo en cuestión usa conceptos de arquitectura tanto del diseño monolítico como del micro-núcleo, específicamente el paso de mensajes y la ejecución de ciertos componentes del sistema operativo en espacio de usuario.

Algunos ejemplos de núcleos híbridos: Microsoft Windows NT, usado en todos los sistemas que usan el código base de Windows NT, XNU (usado en Mac OS X), DragonFlyBSD, ReactOS.

B. POR LOS SERVICIOS QUE OFRECEN

Modo de administración de usuarios o número de usuarios

- ❖ **Monousuario:** el SO solo es utilizado por un usuario, quedando todos los recursos del ordenador de trabajo a su disposición. Ejemplos de estos SO podría ser el MS-DOS.
- ❖ **Multiusuario:** en este caso varios usuarios pueden compartir el ordenador de trabajo, por lo que los recursos del sistema se deben repartir en función de la carga de trabajo que se demande en cada momento. Ejemplos: Unix, Windows Server, Linux, etc.

Modo de administración de tarea o número de procesos

- ❖ **Monotarea:** el Sistema Operativo solo puede gestionar un programa. Es decir, no es capaz de ejecutar varios programas simultáneamente y mucho menos gestionar más de un procesador. El ejemplo más claro sería el MS-DOS.

BLOQUE II

TEMA 04

- ❖ **Multitarea:** en este caso se pueden ejecutar varios programas simultáneamente. Esto no quiere decir que se ejecuten al 100 % en paralelo, puesto que para ello se necesitarían tantos procesadores como programas se ejecuten. Lo que se hace es que el SO intercala los procesos de los programas que se están ejecutando haciendo que parezca que se ejecutan simultáneamente. Con la aparición de los nuevos procesadores de varios núcleos, se facilitan estas tareas y los nuevos sistemas operativos son capaces de aprovechar estos recursos de manera eficiente. Ejemplos: Windows en sus distintas versiones, Unix, Linux, etc.

Por el número de procesadores

- ❖ **Monoprocesador:** un sistema operativo monoprocesador es aquél que es capaz de manejar solamente un procesador del ordenador, de manera que si el ordenador tuviese más de uno le resultaría inútil. Los ejemplos más típicos de este tipo de sistemas son el DOS y el MacOS.
- ❖ **Multiprocesador:** un sistema operativo multiprocesador se aplica en ordenadores con dos o más procesadores, siendo capaz de usarlos todos para distribuir su carga de trabajo. Generalmente estos sistemas trabajan de dos formas: **simétrica o asimétrica**. Cuando se trabaja de manera asimétrica, el sistema operativo selecciona a uno de los procesadores como procesador maestro, el cual servirá como pivote para distribuir la carga a los demás procesadores, que reciben el nombre de esclavos. Cuando se trabaja de manera simétrica, los procesos o partes de ellos (hebras o threads) son enviados indistintamente a cualquiera de los procesadores disponibles, dando lugar bajo este esquema, y teóricamente, a una mejor distribución y un mayor equilibrio en la carga de trabajo

C. POR LA FORMA EN QUE OFRECEN LOS SERVICIOS

- ❖ **Centralizado o en Red:** Aquel que facilita el desarrollo de las actividades en un solo ordenador, siendo esto solamente factible en ordenadores simples y con un solo sistema de aplicación de modo tal que este puede solo desarrollar programas en un equipo determinado.
- ❖ **Descentralizado o Distribuido:** Son aquellos sistemas que se usan en diversos equipos, estos por lo general se revisten en centrales operadoras, es decir, un solo equipo mantiene el programa, pero por medio de extensiones este puede ser manejado en otros dispositivos.

Los sistemas descentralizados permiten manejar las aplicaciones y programas en varios equipos.

BLOQUE II

TEMA 04

2. ELEMENTOS DE UN SISTEMA OPERATIVO

Los componentes básicos de un SO son los siguientes:

- ✓ Administrador de procesos
- ✓ Administrador de memoria principal
- ✓ Administrador del sistema de E/S
- ✓ Administrador de archivos

Estos componentes, llevan a cabo las funciones principales que tiene encomendado un SO:

- ✓ La gestión de procesos.
- ✓ La gestión de memoria.
- ✓ El sistema de archivos.
- ✓ La gestión de la Entrada/Salida.

2.1 GESTION DE PROCESOS

Se puede definir un **proceso** como un programa en ejecución, que ejecuta sentencias, está en un estado concreto en cada momento, y dispone de un conjunto de recursos del ordenador para realizar su tarea: tiempo de CPU, memoria, archivos y dispositivos de E/S.

A la estructura de datos utilizada por los sistemas operativos para almacenar toda la información sobre un proceso (datos, estado, recursos, etc.) se le denomina **Bloque de Control de Procesos (PCB)**. También se conoce como **descriptor de proceso**. Cada proceso tiene un solo PCB. El PCB incluye campos como:

- *Estado del proceso* → El estado actual del proceso.
- *Contador de programa* →Cuál es la siguiente instrucción a ser ejecutada por el proceso.
- *Registros del CPU* → La información específica del estado del CPU mientras el proceso está en ejecución (debe ser respaldada y restaurada cuando se registra un cambio de estado).
- *Información de planificación (scheduling)* → La prioridad del proceso, la cola en que está agendado, y demás información que puede ayudar al sistema operativo a planificar los procesos.
- *Información de administración de memoria* → La información de mapeo de memoria (páginas o segmentos, dependiendo del sistema operativo), incluyendo la pila (stack) de llamadas.

En Linux, al PCB se le llama **Task Struct**

- *Información de contabilidad* → Información de la utilización de recursos que ha tenido este proceso (tiempo total empleado, uso acumulado de memoria y dispositivos, etc.)

BLOQUE II

TEMA 04

- *Estado de E/S* → Listado de dispositivos y archivos asignados que el proceso tiene abiertos en un momento dado.

Cada proceso tiene su contador de programa, registros y variables, aisladas de otros procesos, incluso siendo el mismo programa en ejecución 2 veces.

Un proceso se rige en pequeñas porciones, conocidas como **páginas**, y cada proceso tiene su propia **tabla de paginación**, fungiendo como una optimización del sistema operativo ante los fallos de página.

Esta definición varía ligeramente en el caso de sistemas operativos multihilo, donde un proceso consta de **uno o más hilos**, la memoria de trabajo (compartida por todos los hilos) y la información de planificación. **Cada hilo** consta de instrucciones y estado de ejecución.

Los procesos son creados y eliminados por el sistema operativo, así como también este se debe hacer cargo de la comunicación entre procesos, pero lo hace a petición de otros procesos (interrupción o tiempo de reloj).

SISTEMAS MULTIHILOS

Los hilos son similares a los procesos ya que ambos representan una secuencia simple de instrucciones ejecutada en paralelo con otras secuencias. Los hilos son una forma de dividir un programa en dos o más tareas que corren simultáneamente, compitiendo, en algunos casos, por la CPU. En definitiva, es similar a un proceso, aunque no es lo mismo.

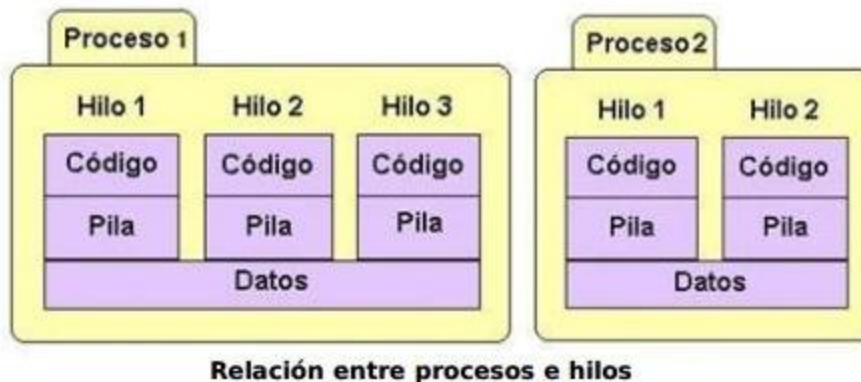
La diferencia más significativa entre los procesos y los hilos, es que los primeros son típicamente independientes, llevan bastante información de estados, e interactúan sólo a través de mecanismos de comunicación dados por el sistema. Por otra parte, los hilos generalmente comparten la memoria, es decir, acceden a las mismas variables globales o dinámicas, por lo que no necesitan costosos mecanismos de comunicación para sincronizarse.

En sistemas operativos que proveen facilidades para los hilos, es **más rápido** cambiar de un hilo a otro dentro del mismo proceso, que cambiar de un proceso a otro.

En los sistemas operativos multihilo es posible crear tanto hilos como procesos. La diferencia estriba en que un proceso solamente puede crear hilos **para sí mismo** y en que dichos hilos comparten toda la memoria reservada para el proceso.

BLOQUE II

TEMA 04



- Si el proceso solo tiene un hilo \equiv Se le llama **proceso clásico** y al hilo se le llama **hilo base**.
- Si el proceso tiene múltiples hilos \equiv Se le llama **proceso moderno** y a cada hilo de ese proceso moderno se le llama **proceso liviano**. Cada hilo tiene una pila diferente ya que cada uno se ejecuta a su propio ritmo.

La unidad de planificación, es decir, que se ejecuta en cada momento, según decida el sistema operativo, no es el proceso moderno, si no los procesos livianos.

PROCESOS	HILOS
Manejados por el S.O.	Manejados por los procesos
Independientes de otros procesos	Relacionados con otros hilos del mismo proceso
Memoria privada, se necesitan mecanismos de comunicación para compartir información	Memoria compartida con el resto de hilos que forman el proceso

BLOQUE II

TEMA 04

2.1.1 BIFURCACION y CLONACIÓN

Ambas operaciones crean un **nuevo proceso** a partir del **proceso padre**, pero de una manera distinta.

El mecanismo por el cual un proceso crea otro proceso, **copia de sí mismo**, se denomina **bifurcación (fork)**, que pasará a actuar como proceso hijo.

Cuando utilizamos la llamada al sistema fork, el proceso hijo creado es una copia exacta del padre (salvo por el PID y la memoria que ocupa). Al proceso hijo se le facilita una copia de las variables del proceso padre y de los descriptores de archivo. Es importante destacar que las variables del proceso hijo son una copia de las del padre (no se refieren físicamente a la misma variable), por lo que modificar una variable en uno de los procesos no se refleja en el otro.

La llamada al sistema clone es mucho más genérica y flexible que el fork, ya que nos permite definir **qué van a compartir** los procesos padre e hijo.

Las llamadas al sistema fork y clone tienen la misma funcionalidad, pero distintas características:

fork \equiv En el momento de la llamada a fork el proceso hijo:

- es una copia exacta del padre excepto el **PID**. (distinto PID entre padre e hijo)
- tiene las mismas variables y archivos abiertos.
- las variables son independientes (padre e hijo tienen distintas zonas de memoria).
- los archivos son compartidos (heredan el descriptor).

clone \equiv Permite especificar qué queremos que compartan padre e hijo.

- espacio de direccionamiento
- información de control del sistema de archivos (file system)
- descriptores de archivos abiertos.
- gestores de señales o PID.

En **Unix/Linux**, una llamada a fork, devuelve el PID del proceso HIJO al proceso PADRE y un 0 al proceso HIJO, o un -1 en el caso de que no se pueda crear el proceso.

BLOQUE II

TEMA 04

2.1.2 CREACION, FINALIZACION Y ESTADOS DE UN PROCESO

Para implementar el modelo de procesos, el Sistema Operativo mantiene una tabla llamada tabla de procesos, con una entrada por proceso. Esta entrada contiene información acerca del estado del proceso y otra información necesaria para restituir el proceso en el punto donde se interrumpió y proseguir su ejecución.

Los principales sucesos que provocan la creación de procesos son los siguientes:

- a) Inicialización del sistema: el arranque de un sistema operativo conlleva la creación de varios procesos, algunos son interactivos o en primer plano, otros son procesos en segundo plano que no están asociados con un usuario en particular y desarrollan alguna función específica.
- b) Ejecución de una llamada al sistema por parte de un proceso en ejecución: el sistema operativo proporciona llamadas para la creación de procesos que pueden ser invocadas por otros procesos.
- c) Solicitud de un usuario para crear un proceso: en los sistemas interactivos, los usuarios pueden iniciar un programa, tecleando un comando o pulsando un icono. Ambas acciones inician un nuevo proceso donde se ejecuta el programa seleccionado.
- d) Inicio de un trabajo por lotes: aplicable a los mainframes con sistemas por lotes. Los usuarios envían una lista de trabajos y el sistema operativo crea los procesos necesarios para ejecutarlos adecuadamente.

TIPOS DE PROCESOS

Existen dos tipos de procesos básicamente, aquellos que se ejecutan en **modo kernel** y aquellos que se ejecutan en **modo usuario**.

Los primeros son más lentos por las llamadas al sistema que realizan, sin embargo, son más seguros por la integridad que representan.

Otra posible clasificación, serían los **procesos en primer plano** y **procesos en segundo plano**.

Los primeros interactúan con el usuario, es decir, el usuario proporciona los datos que el proceso utilizará. Los segundos, son creados para tareas bien definidas y no necesitan la intervención del usuario, por ejemplo, se puede tener un proceso en segundo plano para revisar la temperatura del disco duro constantemente, éstos también son conocidos como **demonios**.

BLOQUE II

TEMA 04

CREACIÓN Y FINALIZACIÓN DE PROCESOS

Respecto a la **creación de procesos**, cuando el SO crea un proceso:

1. Se crea un espacio de direcciones para el proceso (una porción de la memoria)
2. Se crean las estructuras de datos necesarias para conocer la información acerca de ese proceso. A esta estructura se le llama **descriptor del proceso (PCB)**
3. Cuando se termina el proceso, esta estructura se elimina.

El descriptor de proceso es una estructura de datos que está compuesta por la siguiente información:

- Identificador del proceso (PID)
- Relaciones (padre, hijos)
- Propietario. (Usuario dueño del proceso)
- Espacio de direcciones (común a todos los hilos)
- Recursos (Archivos abiertos)
- Referencia a los hilos

Si el proceso tiene hilos, por cada hilo se crea igualmente una estructura de datos denominada **descriptor de hilo**. Compuesta por:

- Identificador del hilo (TPID)
- Relaciones
- Estado (el estado puede ser diferente para cada hilo)
- Estadísticas de ejecución
- Punteros de pila (Cada hilo tiene una pila distinta)
- Recursos específicos
- Contexto

La creación de un proceso puede ser de dos tipos:

Jerárquica → En ella, cada proceso que se crea es hijo del proceso creador y hereda el entorno de ejecución de su padre. El primer proceso que ejecuta un usuario será hijo del intérprete de comandos con el que interactúa.

No jerárquica → Cada proceso creado por otro proceso se ejecuta independientemente de su creador con un entorno diferente. Es un tipo de creación que no suele darse en los sistemas operativos actuales.

Respecto a la **finalización de procesos**, estos pueden terminar por alguno de estos motivos:

1. Salida normal (voluntaria).
2. Salida por error (voluntaria).
3. Error fatal (involuntaria).
4. Terminado por otro proceso (involuntaria).

BLOQUE II

TEMA 04

Otras operaciones sobre procesos pueden ser:

Destruir un proceso → Se trata de la orden de eliminación del proceso con la cual el sistema operativo destruye su PCB.

Suspender un proceso → Es un proceso de alta prioridad que paraliza un proceso que puede ser reanudado posteriormente. Suele utilizarse en ocasiones de mal funcionamiento o sobrecarga del sistema.

Reanudar un proceso → Trata de activar un proceso que ha sido previamente suspendido.

Temporizar la ejecución de un proceso → Hace que un determinado proceso se ejecute cada cierto tiempo (segundos, minutos, horas...) por etapas de una sola vez, pero transcurrido un periodo de tiempo fijo.

Despertar un proceso → Es una forma de desbloquear un proceso que habrá sido bloqueado previamente por temporización o cualquier otra causa.

Cambiar prioridad de un proceso.

Prioridades

Todo proceso por sus características e importancia lleva aparejadas unas determinadas necesidades de ejecución en cuanto a urgencia y asignación de recursos.

Las prioridades según los sistemas operativos se pueden clasificar del siguiente modo:

- *Asignadas por el sistema operativo.* Se trata de prioridades que son asignadas a un proceso en el momento de comenzar su ejecución y dependen fundamentalmente de los privilegios de su propietario y del modo de ejecución.
- *Asignadas por el propietario.*
- *Estáticas.*
- *Dinámicas.*

ESTADOS DE UN PROCESO

Los bloques de control de los procesos (PCB), se almacenan en colas, como se ha comentado anteriormente cada PCB, entre otras informaciones, almacena el estado del proceso.

Los estados de los procesos son internos del SO y transparentes para los usuarios. Se pueden dividir en dos tipos: **activos e inactivos**.

Estados activos ≡ Son aquellos que compiten con el procesador o están en condiciones de hacerlo. Se dividen en:

Ejecución → Estado en el que se encuentra un proceso cuando tiene el control del procesador. En un sistema monoprocesador este estado sólo lo puede tener un proceso.

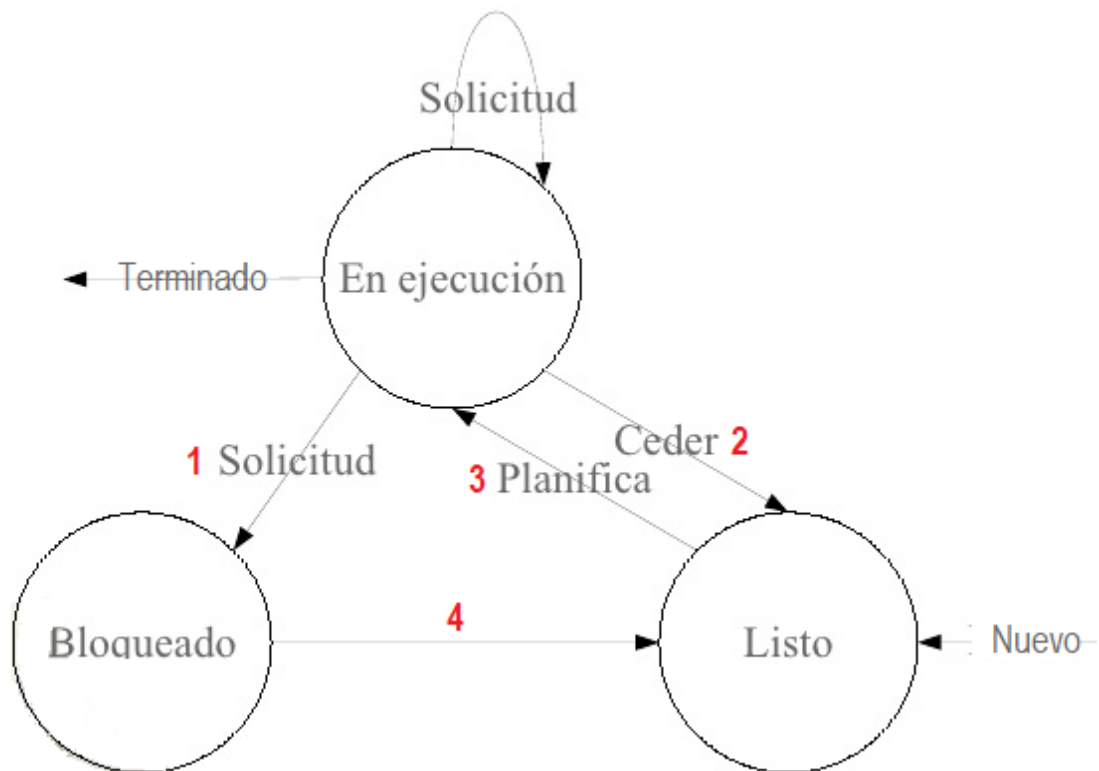
BLOQUE II

TEMA 04

Listo o Preparado → Aquellos procesos que están dispuestos para ser ejecutados, pero no están en ejecución por alguna causa (Interrupción, haber entrado en cola estando otro proceso en ejecución, etc.). Ha dejado disponible al procesador para que otro proceso pueda ocuparlo

Bloqueado → Son los procesos que no pueden ejecutarse de momento por necesitar algún recurso no disponible (generalmente recursos de entrada/salida).

EL diagrama de estados que representa los posibles estados de un proceso y las transiciones entre los mismos es el siguiente:



Hay Cuatro transiciones posibles entre los estados de un proceso:

1. Un proceso se bloquea, por ejemplo, porque está esperando una entrada.
2. El planificador escoge otro proceso.
3. El planificador decide ejecutar este proceso.
4. Cuando las entradas ya han llegado, el proceso está listo para ejecutarse.

En el caso de que uno o varios procesos creen hilos (también llamados subprocesos), estos hilos responden exactamente al mismo diagrama de estado anteriormente reseñado para los procesos.

BLOQUE II

TEMA 04

Un subproceso puede estar en cualquiera de los estados de ejecución de un proceso (En ejecución, Listo, Bloqueado y Terminado) y puede realizar llamadas al sistema para crear nuevos subprocesos.

¿Cuándo son útiles los subprocesos?

- Cuando en diversas aplicaciones se están realizando varias actividades al mismo tiempo, por lo que algunas de ellas podrían bloquearse de vez en cuando.
- Descomponer la aplicación en múltiples subprocesos que se ejecutan casi en paralelo simplifica el modelo de programación.
- Al no estar enlazados con recursos, son más fáciles de crear y destruir que los procesos (aproximadamente 100 veces más rápidos).
- Los subprocesos no mejoran el rendimiento cuando todos hacen un uso intensivo de CPU, pero la mejora es notable cuando el trabajo se reparte equilibradamente entre Proceso y E/S, y aún lo es más en sistemas con múltiples CPUs en los que es posible un verdadero paralelismo.

Estados inactivos \equiv Son aquellos que no pueden competir por el procesador, pero que pueden volver a hacerlo por medio de ciertas operaciones. En estos estados se mantiene el bloque de control de proceso aparcado hasta que vuelva a ser activado. Se trata de procesos que no han terminado su trabajo que lo han impedido y que pueden volver a activarse desde el punto en que se quedaron sin que tengan que volver a ejecutarse desde el principio. Son de dos tipos:

Suspendido bloqueado \rightarrow Es el proceso que fue suspendido en espera de un evento, sin que hayan desaparecido las causas de su bloqueo.

Suspendido programado \rightarrow Es el proceso que han sido suspendido, pero no tiene causa para estar bloqueado.

Otros posibles estados, a tener en cuenta, de un proceso pueden ser:

Nuevo \rightarrow Se solicitó al sistema operativo la creación de un proceso, y sus recursos y estructuras están siendo creados.

Zombie \rightarrow El proceso ha finalizado su ejecución, pero el sistema operativo debe realizar ciertas operaciones de limpieza para poder eliminarlo de la lista.

Terminado \rightarrow El proceso terminó de ejecutarse y sus estructuras están a la espera de ser limpiadas por el sistema operativo.

BLOQUE II

TEMA 04

2.1.3 PLANIFICACIÓN DE PROCESOS

La planificación de procesos se refiere a cómo determina el sistema operativo al orden en que irá cediendo el uso del procesador a los procesos que lo vayan solicitando, y a las políticas que empleará para que el uso que den a dicho tiempo no sea excesivo respecto al uso esperado del sistema.

La planificación de procesos es llevada a cabo por el **planificador o Scheduler**, y debe cumplir los siguientes principios:

- EQUIDAD \equiv Todos los procesos deben ser atendidos
- EFICACIA \equiv El procesador debe estar ocupado el 100% del tiempo
- TIEMPO DE RESPUESTA \equiv El menor posible
- TIEMPO DE REGRESO \equiv Reducir tiempo de espera del resultado
- RENDIMIENTO \equiv Maximizar el ratio N° Tareas/Hora.

TIPOS Y NIVELES DE PLANIFICACIÓN

Existen dos tipos de planificación del procesador:

Planificación no apropiativa \equiv Es aquella en la cual, una vez que a un proceso le toca su turno de ejecución ya no puede ser suspendido, no se le puede arrebatar la CPU. Este esquema puede ser peligroso, ya que, si el proceso contiene ciclos infinitos, el resto de los procesos pueden quedar aplazados indefinidamente.

Planificación apropiativa \equiv Es aquella en que existe un reloj que lanza interrupciones periódicas en las cuales el planificador toma el control y se decide si el mismo proceso seguirá ejecutándose o se le da su turno a otro proceso.

Respecto a los niveles de planificación:

Se puede definir el **scheduling**, traducido como planificación, como el conjunto de políticas y mecanismos construidos dentro del sistema operativo que gobiernan la forma de conseguir que los procesos a ejecutar lleguen a ejecutarse.

El scheduling está muy relacionado con la gestión de los recursos. Existen tres niveles de "scheduler":

Planificación a largo plazo o nivel alto \equiv (**Long Term Scheduler**) Llamada también "planificación de admisión", ya que es la que determina qué trabajos se admiten para su procesamiento y, por consiguiente, se cargan en memoria. (Solo se les admite, no significa que se vayan a ejecutar inmediatamente).

Planificación a medio plazo o nivel medio \equiv (**Mid Term Scheduler**) Decide cuáles procesos es conveniente bloquear en determinado momento, sea por escasez/saturación de algún recurso (como la memoria primaria) o porque están realizando alguna solicitud que no puede satisfacerse momentáneamente. Su misión es mover procesos entre memoria principal y disco (lo que se conoce como **swapping**)

BLOQUE II

TEMA 04

Planificación a corto plazo o nivel bajo \equiv (**dispatcher ó Short Term Scheduler**). Estipula qué procesos en estado preparado pasarán a **ejecución**. Debe ser una planificación sencilla y breve, ya que se ejecutará muchas veces.

Teniendo en cuenta el diagrama de estados anteriormente reflejado, en el que se muestran los estados y transiciones por las que pasa un proceso (o hilo), la relación con estos niveles es la siguiente:

1. El planificador a largo plazo se encarga de admitir un nuevo proceso: la transición de nuevo a listo.
2. El planificador a mediano plazo maneja la activación y bloqueo de un proceso relacionado con eventos, esto es, las transiciones entre en ejecución y bloqueado, y entre bloqueado y listo.
3. El planificador a corto plazo decide entre los procesos que están listos para ejecutarse y determina a cuál de ellos activar, y detiene a aquellos que exceden su tiempo de procesador — implementa las transiciones entre los estados listo y en ejecución.

Los procesos típicamente alternan entre periodos en que realizan principalmente cómputo interno (limitados por CPU, CPU-bound) y otras, en que la atención está puesta en transmitir los datos desde o hacia dispositivos externos (imitados por E/S, I/O-bound). Dado que cuando un proceso se suspende para realizar E/S deja de estar listo (y pasa a estar bloqueado), y desaparece de la atención del planificador a corto plazo, en todo momento los procesos que están en ejecución y listos pueden separarse en:

- *Procesos largos* \rightarrow Aquellos que por mucho tiempo¹ han estado en listos o en ejecución, esto es, procesos que estén en una larga ráfaga limitada por CPU.
- *Procesos cortos* \rightarrow Los que, ya sea que en este momento² estén en una ráfaga limitada por entrada-salida y requieran atención meramente ocasional del procesador, o tienden a estar bloqueados esperando a eventos (como los procesos interactivos).

BLOQUE II

TEMA 04

UNIDADES DE MEDIDA

Desde el punto de vista de la planificación de procesos, en vez de emplear unidades temporales formales (p. ej. Fracciones de segundo), es común emplear ticks y quants. Esto es en buena medida porque, si bien en el campo del cómputo las velocidades de acceso y uso efectivo cambian constantemente, los conceptos y las definiciones permanecen.

Tick \equiv Una fracción de tiempo durante la cual se puede realizar trabajo útil, esto es, usar el CPU sin interrupción. El tiempo correspondiente a un tick está determinado por una señal (interrupción) periódica, emitida por el temporizador (timer). La frecuencia con que ocurre esta señal se establece al inicio del sistema.

NOTA: Jiffies es una variable usada en el sistema operativo Linux que nos indica los ticks que han pasado desde que se arrancó el sistema.

Quantum \equiv El tiempo máximo que se permitirá a un proceso el uso del procesador. Puede ser fijo o variable y puede tener el mismo valor o no para todos los procesos.

Tiempo de espera medio \equiv Promedio de tiempo que los procesos están en estado de "listo"

Tiempo de retorno \equiv Tiempo que transcurre desde que el proceso se lanza hasta que finaliza su ejecución. Se puede ver como la suma del tiempo de espera más el tiempo de ejecución.

Tiempo de retorno medio \equiv Suma de tiempos de retorno de todos los procesos dividida por la cantidad de procesos.

Tiempo de espera \equiv Tiempo que un proceso permanece en la cola de preparados.

Tiempo de respuesta \equiv Tiempo que un proceso bloqueado tarda en entrar en la CPU desde que ocurre el suceso que lo bloquea.

BLOQUE II

TEMA 04

2.1.4 ALGORITMOS DE PLANIFICACIÓN DE PROCESOS

FCFS (First Come First Served) \equiv Primero en llegar primero en ser servido. Este algoritmo emplea una cola de procesos, asignando un lugar a cada proceso por el orden de llegada. Cuando el proceso llega es puesto en su lugar en la cola después del que llegó antes que él y se pone en estado de listo. Cuando un proceso comienza a ejecutarse no se interrumpe su ejecución hasta que termina de hacerlo. El despachador es muy simple, básicamente una cola FIFO.

Si bien un esquema FCFS reduce al mínimo la sobrecarga del procesador, el rendimiento percibido por los últimos procesos en llegar (o por procesos cortos llegados en un momento inconveniente) resulta inaceptable. No favorece ni a los procesos más recientes incluidos en la cola (tienen que esperar a que se ejecuten los que están delante de ellos) ni a los procesos cortos.

SJF (Shortest Job First) \equiv Prioridad al más corto. El proceso que se encuentra en ejecución cambiará de estado voluntariamente, o sea, no tendrá un tiempo de ejecución determinado para el proceso. A cada proceso se le asigna el tiempo que usará cuando vuelva a estar en ejecución, y se irá ejecutando el que tenga un menor tiempo asignado. Si se da el caso de que dos procesos tengan igual valor en ese aspecto emplea el algoritmo FCFS.

Resumiendo, Se ejecutan primero aquellos trabajos que necesitan menos tiempo.

ROUND ROBIN \equiv A cada proceso se le asigna un tiempo determinado para su ejecución (QUANTUM o TIME-SLICE), el mismo **para todos**. En caso de que un proceso no pueda ser ejecutado completamente en ese tiempo se continuará su ejecución después de que todos los procesos restantes sean ejecutados durante el tiempo establecido.

Es un algoritmo basado en FCFS que trata la cola de procesos que se encuentran en estado de listos como una cola circular. Este algoritmo tiene un tiempo de espera relativamente grande. Sin embargo, garantiza un reparto de la CPU entre todos los usuarios y arroja tiempos de respuesta buenos.

ROUND ROBIN ESGOISTA (SRR) \equiv Shelfish Round Robin. Es una variante del algoritmo Round Robin. En este caso se **usan dos colas**, una para los procesos que entran nuevos y otra para los procesos antiguos.

Este método busca favorecer los procesos que ya han pasado tiempo ejecutando en la CPU con respecto a los recién llegados mediante dos colas distintas. De hecho, los nuevos procesos no son programados directamente para su ejecución, sino que se les forma en la cola de procesos nuevos y se avanza únicamente con la cola de procesos aceptados.

BLOQUE II

TEMA 04

POR PRIORIDAD \equiv En este tipo de planificación a cada proceso se le asigna una prioridad siguiendo un criterio determinado, y de acuerdo con esa prioridad será el orden en que se atienda cada proceso.

Si existen varios procesos con la misma prioridad entre ellos pueden ejecutarse por su orden de llegada o por round robin.

La ventaja de este algoritmo es que es flexible en la ejecución de los procesos. Su desventaja es que puede provocar aplazamiento indefinido en los procesos de baja prioridad.

SRT (Shortest Remaining Time) \equiv Tiempo restante más corto. Parecido al del trabajo más corto primero, pero aquí se está calculando en todo momento cuánto tiempo resta para terminar a todos los procesos, incluyendo los nuevos. Aquél más próximo a finalizar es escogido para ejecutarse.

Será capaz de expulsar un proceso largo en ejecución para ejecutar otros más cortos. El problema que puede surgir es que un proceso largo puede llegar a expulsarse muchas veces y nunca terminar debido a la ejecución de otros más cortos.

CFS (Completely Fair Scheduler) \equiv Planificador Completamente Justo. Este algoritmo tiene como objetivo el maximizar el uso de la CPU pero permitiendo el uso interactivo de la máquina. Es decir, tratará de que en ningún momento un usuario vea una bajada de rendimiento.

Con el planificador CFS se realiza un cambio radical en los planteamientos actuales de los planificadores incluidos en Linux, cambiando la planificación de manera que se base en tiempo, en nanosegundos, en vez de colas de ejecución donde hay tareas en espera, sino que se creará un **Árbol rojo-negro** de búsqueda, en el que se almacenará una línea de tiempo de las futuras tareas que usarán la CPU. Ya no usará los jiffies, basado en los tick de la CPU, para expulsar a un proceso.

No usará intervalos de tiempo (quantum) estáticos, sino que se irán modificando dinámicamente según necesidades del sistema.

PLANIFICACIÓN GARANTIZADA \equiv Para realizar esta planificación el sistema tiene en cuenta el número de usuarios que deben ser atendidos. Para un número "n" de usuarios se asignará a cada uno un tiempo de ejecución igual a $1/n$.

MLQ (Colas Múltiples o multinivel) \equiv Multilevel Queue. En este algoritmo la cola de procesos que se encuentran en estado de listos es dividida en un número determinado de colas más pequeñas. Los procesos son clasificados mediante un criterio para determinar en qué cola será colocado cada uno cuando quede en estado de listo. Cada cola puede manejar un algoritmo de planificación diferente a las demás y no permiten el movimiento de los procesos entre las distintas colas.

Existe una variante, llamada **colas multinivel realimentadas** (MLFQ multi level feedback queues), que se basan en los algoritmos de colas multinivel, pero permiten el movimiento de los trabajos de unas colas a otras.

BLOQUE II

TEMA 04

2.1.5 CONCURRENCIA

Dos procesos son concurrentes cuando se ejecutan de manera que sus intervalos de ejecución se solapan.



En los sistemas de tiempo compartido se presentan muchos problemas debido a que los procesos compiten por los recursos del sistema que son únicos. Existe una multitud de recursos cuyo acceso debe ser controlado para evitar los problemas de la concurrencia.

Los principales problemas de concurrencia son los siguientes.

Condiciones de carrera o competencia \equiv La condición de carrera (race condition) ocurre cuando dos o más procesos acceden a un recurso compartido sin control, de manera que el resultado combinado de este acceso depende del orden de llegada.

Postergación o aplazamiento indefinido \equiv Consiste en el hecho de que uno o varios procesos nunca reciban el suficiente tiempo de ejecución para terminar su tarea. Por ejemplo, que un proceso ocupe un recurso y lo marque como ocupado y que termine sin marcarlo como desocupado. Si algún otro proceso pide ese recurso, lo verá ocupado y esperará indefinidamente a que se desocupe.

Condición de espera circular \equiv Se da cuando dos o más procesos forman una cadena de espera que los involucra a todos. Por ejemplo, supongamos que el proceso A tiene asignado el recurso cinta y el proceso B tiene asignado el recurso disco. Si el proceso A solicita el recurso disco y el proceso B solicita el recurso cinta, se forma una espera circular entre ambos procesos que se puede evitar quitándole a la fuerza un recurso a cualquiera de los dos procesos.

Condición de no apropiación \equiv Esta condición no resulta precisamente de la concurrencia, pero juega un papel importante en este ambiente. Especifica que, si un proceso tiene asignado un recurso, dicho recurso no puede arrebatársele por ningún motivo, y estará disponible para dicho proceso hasta que éste suelte por su propia voluntad.

Condición de espera ocupada \equiv Consiste en que un proceso pide un recurso que ya está asignado a otro proceso, debiendo cumplir la condición de no apropiación. En ese caso, el proceso estará gastando el resto de su time-slice chequeando si el recurso fue liberado. Es decir, desperdicia su tiempo de ejecución en esperar. La solución más común a este problema viene dada porque el sistema operativo se dé cuenta de esta situación y mande el proceso a una cola de espera, otorgándole inmediatamente el turno de ejecución a otro proceso.

Condición de exclusión mutua \equiv Cuando un proceso utiliza un recurso del sistema realiza una serie de operaciones sobre el recurso, y después lo deja de usar. A la sección de código que usa ese recurso se le llama **región crítica**.

BLOQUE II

TEMA 04

La condición de exclusión mutua establece que solamente se permite a un único proceso estar dentro de la misma región crítica. Esto es, en cualquier momento solamente un proceso puede usar un recurso a la vez.

Condición de ocupar y esperar un recurso \equiv Consiste en que un proceso pide un recurso y se le asigna. Antes de soltarlo, pide otro recurso que otro proceso ya tiene asignado.

Deadlock y Livelock

En sistemas operativos, el bloqueo mutuo (también conocido como interbloqueo, traba mortal, **deadlock**, abrazo mortal) es el bloqueo permanente de un conjunto de procesos o hilos de ejecución en un sistema concurrente que compiten por recursos del sistema o bien se comunican entre ellos.

En la vida real, un ejemplo puede ser el de dos niños que intentan jugar al arco y flecha, uno toma el arco, el otro la flecha. Ninguno puede jugar hasta que alguno libere lo que tomó.

Se presentan al mismo tiempo cuatro condiciones necesarias: la condición de no apropiación, la condición de espera circular, la condición de exclusión mutua y la condición de ocupar y esperar un recurso. (Conocidas como condiciones de Coffman).

Los bloqueos mutuos pueden ser evitados si se sabe cierta información sobre los procesos antes de la asignación de recursos. Los bloqueos mutuos pueden prevenirse asegurando que no suceda alguna de las condiciones necesarias vistas anteriormente.

Existen varios algoritmos para evitar bloqueos mutuos:

- Algoritmo del banquero, introducido por Dijkstra.
- Algoritmo de grafo de asignación de recursos.
- Algoritmo de Seguridad.
- Algoritmo de solicitud de recursos.

Un **livelock** es similar a un deadlock, excepto que el estado de los dos procesos envueltos en el livelock constantemente cambia con respecto al otro. Livelock es una forma de inanición y la definición general solo dice que un proceso específico no está procesando.

En un ejemplo del mundo real, un livelock ocurre por ejemplo cuando dos personas, al encontrarse en un pasillo angosto avanzando en sentidos opuestos, y cada una trata de ser amable moviéndose a un lado para dejar a la otra persona pasar, pero terminan moviéndose de lado a lado sin tener ningún progreso, pues ambos se mueven hacia el mismo lado, al mismo tiempo.

BLOQUE II

TEMA 04

2.2 GESTIÓN DE LA MEMORIA

La parte del sistema operativo que administra la memoria se llama administrador de memoria y su labor consiste en llevar un registro de las partes de memoria que se estén utilizando y aquellas que no, con el fin de asignar espacio en memoria a los procesos cuando éstos la necesiten y liberándola cuando terminen.

La administración de memoria se refiere a los distintos métodos y operaciones que se encargan de obtener la máxima utilidad de la memoria, organizando los procesos y programas que se ejecutan de manera tal que se aproveche de la mejor manera posible el espacio disponible.

2.2.1 LA UNIDAD DE MANEJO DE MEMORIA (MMU)

Con la introducción de sistemas multitarea, es decir, dos o más programas ejecutándose, se vio la necesidad de tener más de un programa cargado en memoria. Esto conlleva que el sistema operativo junto con información del programa a ejecutar debe resolver cómo ubicar los programas en la memoria física disponible. Hoy en día, además, casi todos los sistemas operativos emplean implementaciones que requieren de **hardware** especializado: la Unidad de Manejo de Memoria (MMU).

Permite emplear más memoria de la que está directamente disponible, con el propósito de ofrecer a los procesos más espacio de lo que puede direccionar la arquitectura (hardware) empleada.

La MMU es también la encargada de verificar que un proceso no tenga acceso a leer o modificar los datos de otro, liberando al procesador de tener que hacer esa tarea (aumenta el rendimiento).

2.2.2 ASIGNACIÓN DE MEMORIA CONTIGUA

Las técnicas que existen para la carga de programas en la memoria son:

Partición fija → que es la división de la memoria libre en varias partes (de igual o distinto tamaño). Dichas particiones se crean cuando se enciende el equipo, y permanecen con los tamaños iniciales hasta que el equipo se reinicia.

Partición dinámica → que son las particiones de la memoria en tamaños que pueden ser variables, según la cantidad de memoria que necesita cada proceso. Para esto, el sistema debía mantener ya una estructura de datos suficiente para saber en dónde había huecos disponibles de RAM y de dónde a dónde había particiones ocupadas por programas en ejecución. Así, cuando un programa requería ser cargado a RAM, el sistema analizaba los huecos para saber si había alguno de tamaño suficiente para el programa entrante, si era así, le asignaba el espacio. Si no, intentaba relocalizar los programas existentes con el propósito de hacer contiguo todo el espacio libre y así obtener un hueco de tamaño suficiente. Si aun así el espacio era insuficiente, el programa se bloqueaba y el sistema optaba por otro. Al proceso mediante el que se juntan los huecos se le denomina de **compactación**.

BLOQUE II

TEMA 04

Fragmentación

Comienza a aparecer cuando más procesos terminan su ejecución, y el sistema operativo libera la memoria asignada a cada uno de ellos. A medida que los procesos finalizan, aparecen "huecos" de memoria disponible, seguidas por regiones de memoria usada por los procesos que aún se encuentran activos.

Si la computadora no tiene hardware específico que permita que los procesos resuelvan sus direcciones en tiempo de ejecución, el sistema operativo no puede reasignar los bloques existentes, y aunque pudiera hacerlo, mover un proceso entero en memoria puede resultar una operación costosa en tiempo de procesamiento.

Al crear un nuevo proceso, el sistema operativo tiene tres estrategias según las cuales podría asignarle uno de los bloques (huecos) disponibles (conocidas como estrategias de Colocación):

Primer ajuste \equiv El sistema toma el **primer** bloque con el tamaño **suficiente** (igual o mayor) para alojar el nuevo proceso. Este es el mecanismo más simple de implementar y el de más rápida ejecución. No obstante, esta estrategia puede causar el desperdicio de memoria, si el bloque no es exactamente del tamaño requerido.

Mejor ajuste \equiv El sistema busca entre todos los bloques disponibles cuál es el que mejor se ajusta al tamaño requerido por el nuevo proceso. Esto implica la revisión completa de la lista de bloques, pero permite que los bloques remanentes, una vez que se ubicó al nuevo proceso, sean tan pequeños como sea posible (esto es, que haya de hecho un mejor ajuste).

Peor ajuste \equiv El sistema busca cuál es el bloque más grande disponible, y se lo asigna al nuevo proceso. Al igual que en el caso anterior, requiere una búsqueda exhaustiva de hueco a asignar.

Siguiente ajuste \equiv Es igual que el primer ajuste con la diferencia que se deja un apuntador al lugar en donde se asignó el último hueco para realizar la siguiente búsqueda a partir de él.

Ajuste rápido \equiv Se mantienen listas enlazadas separadas de acuerdo a los tamaños de los huecos, para así buscarle a los procesos un hueco más rápido en la cola correspondiente.

BLOQUE II

TEMA 04

La fragmentación que se produce puede ser de dos tipos:

Fragmentación interna \equiv Generada cuando se reserva más memoria de la que el proceso va realmente a usar. Sin embargo, a diferencia de la externa, estos huecos no se pueden compactar para ser utilizados. Se debe de esperar a la finalización del proceso para que se libere el bloque completo de la memoria.

Fragmentación externa \equiv Se produce cuando los procesos asignados han ocupado posiciones no contiguas de memoria dejando demasiados bloques libres de pequeño tamaño, en los que no "caben" nuevos procesos.

En la memoria principal se soluciona compactando los procesos para que estos ocupen posiciones contiguas y dejar los bloques libres juntos, o también se soluciona con la paginación de memoria.

Tanto el primer ajuste, como el mejor y el peor producen **fragmentación externa**

Compactación

Un problema importante que va surgiendo como resultado de esta fragmentación es que el espacio total libre de memoria puede ser mucho mayor que lo que requiere un nuevo proceso, pero al estar fragmentada en muchos bloques, éste no encontrará una partición contigua donde ser cargado.

Si los procesos emplean resolución de direcciones en tiempo de ejecución, cuando el sistema operativo comience a detectar un alto índice de fragmentación, puede lanzar una operación de compresión o compactación. Esta operación consiste en mover los contenidos en memoria de los bloques asignados para que ocupen espacios contiguos, permitiendo unificar varios bloques libres contiguos en uno solo.

La compactación tiene un costo alto —involucra mover prácticamente la totalidad de la memoria (probablemente más de una vez por bloque).

Intercambio (swap / swapping) con el almacenamiento secundario

Siguiendo de cierto modo la lógica requerida por la compactación se encuentran los sistemas que utilizan intercambio (swap) entre la memoria primaria y secundaria. En éstos, el sistema operativo puede comprometer más espacio de memoria del que tiene físicamente disponible. Cuando la memoria se acaba, el sistema suspende un proceso (usualmente un proceso "bloqueado") y almacena una copia de su imagen en memoria en almacenamiento secundario (disco duro) para luego poder restaurarlo.

La porción de disco duro a usar como zona de swap, puede tener dos ubicaciones dentro del disco duro:

Fichero de Intercambio o paginación \rightarrow Se trata de un único fichero que almacena la información necesaria para realizar la función de swap. Es sencillo de crear, borrar, etc. El principal problema es que al ser un fichero más del disco duro, le afecta la fragmentación del mismo.

BLOQUE II

TEMA 04

Partición de Intercambio → Se trata de usar una partición completa del disco duro como zona de swap. Tiene un mejor rendimiento si se coloca esta partición al principio del disco. Entre las ventajas que presenta es que no se ve afectada por la fragmentación y que no es necesario usar un sistema de ficheros concreto.

2.2.3 PAGINACIÓN

Los sistemas de paginación de memoria usados por los sistemas operativos, dividen los programas en pequeñas partes llamadas **páginas**.

Del mismo modo, la memoria es dividida en trozos del mismo tamaño que las páginas llamados **marcos de página**.

De esta forma, la cantidad de memoria desperdiciada por un proceso es el final de su última página, lo que:

- **Minimiza** la fragmentación interna
- **Evita** la externa.

En un momento cualquiera, la memoria se encuentra ocupada con páginas de diferentes procesos, mientras que algunos marcos están disponibles para su uso. El sistema operativo mantiene una lista de estos últimos marcos, y una tabla por cada proceso, donde consta en qué marco se encuentra cada página del proceso. De esta forma, las páginas de un proceso pueden no estar continuamente ubicadas en memoria, y pueden intercalarse con las páginas de otros procesos.

Las **tablas de paginación o tablas de páginas** son una parte integral del Sistema de Memoria Virtual en sistemas operativos, cuando se utiliza paginación. Son usadas para realizar las traducciones de direcciones de memoria virtual (o lógica) a memoria real (o física) y en general el sistema operativo mantiene una por cada proceso corriendo en el sistema.

En la tabla de páginas de un proceso, se encuentra la ubicación del marco que contiene a cada una de sus páginas. Las direcciones lógicas ahora se forman como un número de página y de un desplazamiento dentro de esa página (conocido comúnmente como offset). El número de página es usado como un índice dentro de la tabla de páginas, y una vez obtenida la dirección del marco de memoria, se utiliza el desplazamiento para componer la dirección real o dirección física.

Tabla de páginas invertida

Técnica de paginación en la cual hay una entrada por cada página real de la memoria e incluye información del proceso que posee dicha página. En el sistema solo habrá una tabla de páginas invertida y esta solo tendrá una entrada por cada "marco" en la memoria física. La ventaja de este método es que reduce la memoria física ocupada por la tabla de páginas. Por otra parte, aumenta el tiempo de búsqueda de páginas ya que se debe explorar la tabla cada vez que hay una referencia a una página, debido a que esta se encuentra ordenada según la memoria física y las búsquedas se realizan según la memoria virtual.

BLOQUE II

TEMA 04

Hiperpaginación

Se denomina hiperpaginación (**thrashing**), a la situación en la que se utiliza una creciente cantidad de recursos para hacer una cantidad de trabajo cada vez menor.

Las formas de evitar la hiperpaginación se basan en la idea de intentar adivinar qué páginas serán utilizadas próximamente, basados en su historia reciente y utilizando como hipótesis el principio de cercanía de referencias. Estos son los denominados algoritmos de reemplazo de páginas.

Desde un punto de vista más práctico, se puede evitar la hiperpaginación:

1. Aumentando la cantidad de memoria RAM (generalmente la mejor solución a largo plazo).
2. Disminuyendo la cantidad de aplicaciones corriendo en la computadora.
3. Ajustando el tamaño de la partición de intercambio.

2.2.4 SEGMENTACIÓN

Al desarrollar un programa en un lenguaje de alto nivel, el programador usualmente no se preocupa por la ubicación en la memoria física de los diferentes elementos que lo componen. Esto se debe a que en estos lenguajes las variables y funciones son referenciadas por sus nombres, no por su ubicación.

No obstante, cuando se compila el programa para una arquitectura que soporte segmentación, el compilador ubicará a cada una de las secciones en un segmento diferente.

Esto permite activar los mecanismos que evitan la escritura accidental de las secciones de memoria del proceso que no se deberían modificar (aquellas que contienen código o de sólo lectura), y permitir la escritura de aquellas que sí (en las cuales se encuentran las variables globales, la pila o stack y el espacio de asignación dinámica o heap).

Así, los elementos que conforman un programa se organizan en secciones: una sección contiene el espacio para las variables globales, otra sección contiene el código compilado, otra sección contiene la tabla de símbolos, etc.

Luego, cuando el sistema operativo crea un proceso a partir del programa, debe organizar el contenido del archivo ejecutable en memoria. Para ello carga en memoria algunas secciones del archivo ejecutable (como mínimo la sección para las variables globales y la sección de código) y puede configurar otras secciones como la pila o la sección de libres. Para garantizar la protección de cada una de estas secciones en la memoria del proceso, el sistema puede definir que cada sección del programa se encuentra en un segmento diferente, con diferentes tipos de acceso.

BLOQUE II

TEMA 04

Características:

- Permite separar las regiones de la memoria lineal en segmentos, cada uno de los cuales puede tener **diferentes permisos de acceso**.
- Ayuda a incrementar la modularidad de un programa: es muy común que las bibliotecas ligadas dinámicamente estén representadas en segmentos independientes.
- Permite **reducir la fragmentación interna** de la memoria provocada por la paginación, ya que asigna a cada programa la cantidad de memoria que requiere.

2.2.5 MEMORIA VIRTUAL

Es una técnica de gestión de la memoria que se encarga de que el sistema operativo disponga, tanto para el software de usuario como para sí mismo, de mayor cantidad de memoria que esté disponible físicamente. La mayoría de los ordenadores tienen cuatro tipos de memoria: registros en la CPU, la memoria caché (tanto dentro como fuera del CPU), la memoria RAM y el disco duro. En ese orden, van de menor capacidad y mayor velocidad a mayor capacidad y menor velocidad.

Muchas aplicaciones requieren acceso a más información (código y datos) que la que se puede mantener en memoria física. Esto es así sobre todo cuando el sistema operativo permite múltiples procesos y aplicaciones ejecutándose simultáneamente. Una solución al problema de necesitar mayor cantidad de memoria de la que se posee consiste en que las aplicaciones mantengan parte de su información en disco, moviéndola a la memoria principal cuando sea necesario. Hay varias formas de hacer esto.

Una opción es que la aplicación misma sea responsable de decidir qué información será guardada en cada sitio (segmentación), y de traerla y llevarla. La desventaja de esto, además de la dificultad en el diseño e implementación del programa, es que es muy probable que los intereses sobre la memoria de dos o varios programas generen conflictos entre sí: cada programador podría realizar su diseño teniendo en cuenta que es el único programa ejecutándose en el sistema. La alternativa es usar memoria virtual, donde la combinación entre hardware especial y el sistema operativo hace uso de la memoria principal y la secundaria para hacer parecer que el ordenador tiene mucha más memoria principal (RAM) que la que realmente posee. Este método es **invisible** a los procesos.

Aunque la memoria virtual podría estar implementada por el software del sistema operativo, en la práctica casi siempre se usa una combinación de hardware y software, dado el esfuerzo extra que implicaría para el procesador.

BLOQUE II

TEMA 04

Cuando se usa memoria virtual, o cuando una dirección es leída o escrita por la CPU, una parte del hardware dentro de la computadora traduce las direcciones de memoria generadas por el software (direcciones virtuales) en:

- La dirección real de memoria (la dirección de memoria física).
- Una indicación de que la dirección de memoria deseada no se encuentra en memoria principal (**excepción de memoria virtual o Fallo de página**)

La traducción de las direcciones virtuales a reales es implementada por una Unidad de Manejo de Memoria (MMU). El sistema operativo es el responsable de decidir qué partes de la memoria del programa es mantenida en memoria física. Además, mantiene las tablas de traducción de direcciones (si se usa paginación la tabla se denomina tabla de paginación), que proveen las relaciones entre direcciones virtuales y físicas, para uso de la MMU.

Finalmente, cuando una excepción de memoria virtual ocurre (Fallo de página), el sistema operativo es responsable de ubicar un área de memoria física para guardar la información faltante, trayendo la información desde el disco, actualizando las tablas de traducción y finalmente continuando la ejecución del programa que dio la excepción de memoria virtual desde la instrucción que causó el fallo.

Paginación en memoria virtual

La memoria virtual usualmente (pero no de forma imprescindible) es implementada usando paginación.

El único inconveniente de paginación pura es que todas las páginas de un proceso deben estar en memoria para que se pueda ejecutar. Esto hace que, si los programas son de tamaño considerable, no puedan cargarse muchos a la vez, disminuyendo el grado de multiprogramación del sistema. Para evitar esto, y aprovechando el principio de cercanía de referencias donde se puede esperar que un programa trabaje con un conjunto cercano de referencias a memoria (es decir con un conjunto residente más pequeño que el total de sus páginas), se permitirá que algunas páginas del proceso sean guardadas en un espacio de intercambio (fragmentación interna) mientras no se necesiten.

Fallos de página

En un sistema de memoria virtual paginada, un fallo de página (page fault) es una excepción arrojada cuando un programa informático requiere una dirección que no se encuentra en **la memoria principal** actualmente. Aunque el término sugiere un mal funcionamiento, se trata de un procedimiento normal dentro de la marcha del programa.

BLOQUE II

TEMA 04

2.3 SISTEMA DE ARCHIVOS

Un sistema de archivos (File System, abreviado como FS) controla cómo se almacenan y recuperan los datos. Sin un sistema de archivos, los datos colocados en un medio de almacenamiento serían un gran cuerpo de datos sin forma de saber dónde termina un dato y comienza el siguiente.

La estructura y las reglas lógicas que se utilizan para administrar los grupos de datos y sus nombres se denomina "sistema de archivos".

Un FS consta de dos o tres capas. A veces, las capas se separan explícitamente y, a veces, las funciones se combinan

La primera capa, es el **sistema de archivos lógico**, responsable de la interacción con la aplicación del usuario. Proporciona la interfaz de programa de aplicación (API) para las operaciones de archivo (OPEN, CLOSE, READ, etc.), y pasa a la operación solicitada a la capa de debajo de ella para su procesamiento.

El sistema de archivos lógico administra las entradas de la tabla de archivos abiertos y los descriptores de archivos por proceso. Esta capa proporciona acceso a archivos, operaciones de directorio, seguridad y protección.

La segunda capa, opcional, es el **sistema de archivos virtual**. Esta interfaz permite la compatibilidad con varias instancias simultáneas de sistemas de archivos físicos.

La tercera capa es el **sistema de archivos físico**. Esta capa se ocupa del funcionamiento físico del dispositivo de almacenamiento (por ejemplo, disco). Procesa bloques físicos que se leen o escriben. Maneja el almacenamiento en búfer y la gestión de la memoria y es responsable de la ubicación física de los bloques en ubicaciones específicas del medio de almacenamiento.

2.3.1 CONCEPTOS PREVIOS

Partición ≡ Una subdivisión de un disco, por medio de la cual el administrador o usuario del sistema puede definir la forma en que se emplea el espacio del disco, segmentándolo según haga falta. Un disco puede tener varias particiones, y cada una de ellas puede tener un FS independiente.

Volumen ≡ Colección de bloques inicializados con un sistema de archivos que pueden presentarse al usuario como una unidad. Típicamente un volumen coincide con una partición, pero no siempre tiene que ser así.

El volumen se describe ante el sistema operativo en el bloque de control de volumen, también conocido como supe bloque en Unix, o tabla maestra de archivos (master file table) en NTFS.

BLOQUE II

TEMA 04

Sistema de archivos \equiv Esquema de organización que sigue un determinado volumen. Dependiendo del sistema de archivos elegido, cada uno de los componentes aquí presentados ocuparán un distinto lugar en el disco, presentando una semántica propia.

Para poder tener acceso a la información almacenada en determinado volumen, el sistema operativo debe tener soporte para el sistema de archivos particular en que éste esté estructurado.

Directorio raíz \equiv La estructura que relaciona cada nombre de archivo con sus números de i-nodo. Típicamente sólo almacena los archivos que están en el primer nivel jerárquico del sistema, y los directorios derivados son únicamente referenciados desde éste.

En sistemas de archivos modernos, el directorio normalmente incluye sólo el nombre de cada uno de los archivos y el número de i-nodo que lo describe, todos los metadatos adicionales están en los respectivos i-nodos.

Metadatos \equiv Recibe este nombre toda la información acerca de un archivo que no es el contenido del archivo mismo. Por ejemplo, el nombre, tamaño o tipo del archivo, su propietario, el control de acceso, sus fechas de creación, último acceso y modificación, ubicación en disco, etc.

I-nodo \equiv (i-node, information node) En los sistemas tipo Windows, normalmente se le denomina **bloque de control de archivo (FCB)**. Es la estructura en disco que guarda los metadatos de cada uno de los archivos, proporcionando un vínculo entre la entrada en el directorio y los datos que lo conforman.

La información almacenada incluye todos los metadatos relacionados con el archivo a excepción del nombre: los permisos y propietarios del archivo, sus fechas de creación, última modificación y último acceso, y la relación de bloques que ocupa en el disco.

Se explica con mayor profundidad más adelante, dentro del apartado dedicado a Unix/Linux.

Mapa de bits de espacio libre \equiv La función del bitmap es poder gestionar el espacio libre del disco

BLOQUE II

TEMA 04

2.3.2 ESQUEMAS DE ASIGNACIÓN DE ESPACIO DE ALMACENAMIENTO

El subsistema de archivos se debe encargar de localizar espacio libre en los medios de almacenamiento para guardar archivos y para después borrarlos, renombrarlos o agrandarlos. Para ello se vale de ubicaciones especiales que contienen la lista de archivos creados y por cada archivo una serie de direcciones que apuntan a su contenido. Esas localidades especiales se llaman directorios. Para asignarle espacio a los archivos existen tres criterios generales que se describen a continuación.

ASIGNACIÓN CONTIGUA \equiv Cada fichero ocupa bloques con direcciones lógicas del dispositivo contiguas \Rightarrow bloques contiguos en el disco. Número de búsquedas y tiempo de búsqueda mínimos.

Asignación contigua definida por:

- Dirección en disco del bloque inicial.
- Longitud del área asignada al archivo (nº de bloques).

Permite el acceso secuencial y directo.

Se produce fragmentación externa, ya que al borrar un fichero queda un hueco que puede no ser utilizado completamente por otro fichero, dado que el tamaño del fichero es más grande que el del bloque que ha quedado libre. Se soluciona realizando una compactación (creación de un único hueco suficientemente grande).

RESUMEN:

- Se necesita la dirección del primer bloque.
- Todo el archivo se puede leer de una sola vez (buen rendimiento)
- El método no es realizable, a menos que se conozca el tamaño máximo del archivo, en el momento de su creación.
- Produce bastante fragmentación externa.

ASIGNACIÓN LIGADA O ENCADENADA \equiv Con este criterio los directorios contienen los nombres de archivos y por cada uno de ellos la dirección del bloque inicial que compone al archivo.

Un archivo se ve como una lista enlazada de bloques de disco. Los bloques pueden estar dispersos por todo el disco. La entrada al directorio contiene un puntero al primer y al último bloque del archivo.

Salto entre bloques \Rightarrow cada bloque contiene un puntero al siguiente bloque.

Se solucionan los problemas de la asignación contigua:

- No hay fragmentación externa ya que no hay necesidad de declarar el tamaño de un archivo en el momento de crearlo.
- Un archivo puede continuar creciendo siempre que haya bloques libres.
- No es necesario compactar el disco.

BLOQUE II

TEMA 04

Sólo es eficiente para archivos de acceso secuencial. Se desperdicia el espacio ocupado por los punteros (se utiliza espacio para guardar punteros perdiéndolo para guardar información). Crece la fragmentación interna, pues se desperdicia más espacio cuando un clúster está parcialmente lleno que cuando un bloque está parcialmente lleno.

ASIGNACIÓN INDEXADA \equiv En este esquema se guarda en el directorio un bloque de índices para cada archivo, con apuntadores hacia todos sus bloques constituyentes, de manera que el acceso directo se agiliza notablemente, a cambio de sacrificar varios bloques para almacenar dichos apuntadores.

Cada archivo tiene su propio bloque índice.

La i-ésima entrada del bloque índice apunta al i-ésimo bloque del archivo.

El directorio contiene la dirección del bloque índice.

Soporta el acceso directo sin sufrir fragmentación externa.

Como inconveniente, desperdicia espacio (gasto de los punteros del bloque índice).

COMPARACION ENTRE MÉTODOS

TIPO	VENTAJAS	INCONVENIENTES
CONTIGUA	Soporta acceso secuencial y directo. Todo el espacio se utiliza para almacenar datos	Encontrar espacio para la creación de un fichero. Fragmentación externa Declaración por anticipado del tamaño del archivo
ENLAZADA	No se produce fragmentación externa No es necesario declarar por anticipado el tamaño del archivo	Eficiente sólo para archivos de acceso secuencial. Espacio ocupado por los punteros. Confiabilidad: Pérdida de datos provocada por pérdida de punteros.
INDEXADA	No se produce fragmentación externa. No es necesario declarar por anticipado el tamaño del archivo. Soporta acceso secuencial y directo.	Mayor pérdida de espacio (bloque/s índice) Confiabilidad: Pérdida de datos provocada por pérdida del bloque/s índice.

BLOQUE II

TEMA 04

2. SISTEMAS UNIX Y LINUX

2.1 INTRODUCCIÓN

¿Qué es UNIX?

El sistema operativo UNIX nació a finales de la década de 1960. AT&T Bell Labs lanzó un sistema operativo llamado Unix escrito en C, que permite una modificación, aceptación y portabilidad más rápidas. Pasó a convertirse en los sistemas operativos más utilizados. Unix es un sistema operativo **propietario**.

El sistema operativo Unix funciona en CLI (Command Line Interface), pero recientemente, ha habido desarrollos para GUI en sistemas Unix. Unix es un sistema operativo que es popular en empresas, universidades, grandes empresas, etc.

¿Qué es LINUX?

Linux es un sistema operativo construido por Linus Torvalds en la Universidad de Helsinki en 1991. El nombre "Linux" proviene del kernel de Linux.

El desarrollo de Linux es uno de los ejemplos más destacados de colaboración de **software libre y de código abierto**. Hoy en día, muchas empresas y un número similar de personas han lanzado su propia versión del sistema operativo basada en el kernel de Linux.

DIFERENCIA CLAVE

- El código fuente de Linux está disponible para el público en general (software Libre), mientras que en Unix el código fuente es propietario.
- UNIX se creó a finales de la década de 1960 en AT&T Bell Labs. Linux fue creado por Linus Torvalds en la Universidad de Helsinki en 1991
- La principal diferencia entre Linux y Unix es que Linux es un clon de Unix
- El Shell predeterminado de Linux es BASH mientras que el Shell de Unix es Bourne Shell.
- Una diferencia clave entre Unix y Linux es que la detección de amenazas y la solución de Linux son muy rápidas, mientras que los usuarios de Unix requieren tiempos de espera más largos para obtener el parche de corrección de errores adecuado.

BLOQUE II

TEMA 04

Diferencias clave	Linux	Unix
Costo	Linux se distribuye libremente, se descarga a través de revistas, libros, sitios web, etc. También hay versiones de pago disponibles para Linux.	Los diferentes sabores de Unix tienen diferentes precios según el tipo de proveedor.
Desarrollo	Linux es de código abierto y miles de programadores colaboran en línea y contribuyen a su desarrollo.	Los sistemas Unix tienen diferentes versiones. Estas versiones son desarrolladas principalmente por AT&T, así como por otros proveedores comerciales.
Usuario	Todos. Desde usuarios domésticos hasta desarrolladores y entusiastas de la informática.	El UNIX se puede utilizar en servidores de Internet, estaciones de trabajo y PC.
Interfaz de texto	BASH es el Shell predeterminado de Linux. Ofrece soporte para múltiples intérpretes de comandos.	Fabricado originalmente para trabajar en Bourne Shell. Sin embargo, ahora es compatible con muchos otros programas.
GUI	Linux proporciona dos GUI, a saber, KDE y Gnome. Aunque existen muchas alternativas como Mate, LXDE, Xfce, etc.	Common Desktop Environment y también tiene Gnome.
Virus	Linux ha tenido alrededor de 60-100 virus listados hasta la fecha que actualmente no se están propagando.	Hay entre 80 y 120 virus reportados hasta la fecha en Unix.
Detección de amenazas	La detección y solución de amenazas es muy rápida porque Linux se basa principalmente en la comunidad. Entonces, si algún usuario de Linux publica algún tipo de amenaza, un equipo de desarrolladores calificados comienza a trabajar para resolver esta amenaza.	Los usuarios de Unix requieren más tiempo de espera para obtener el parche de corrección de errores adecuado.
Arquitecturas	Desarrollado inicialmente para procesadores de hardware x86 de Intel. Está disponible para más de veinte tipos diferentes de CPU que también incluyen un ARM.	Está disponible en máquinas PA-RISC e Itanium.
Uso	El sistema operativo Linux se puede instalar en varios tipos de dispositivos como dispositivos móviles y tabletas.	El sistema operativo UNIX se utiliza para servidores de Internet, estaciones de trabajo y PC.

BLOQUE II

TEMA 04

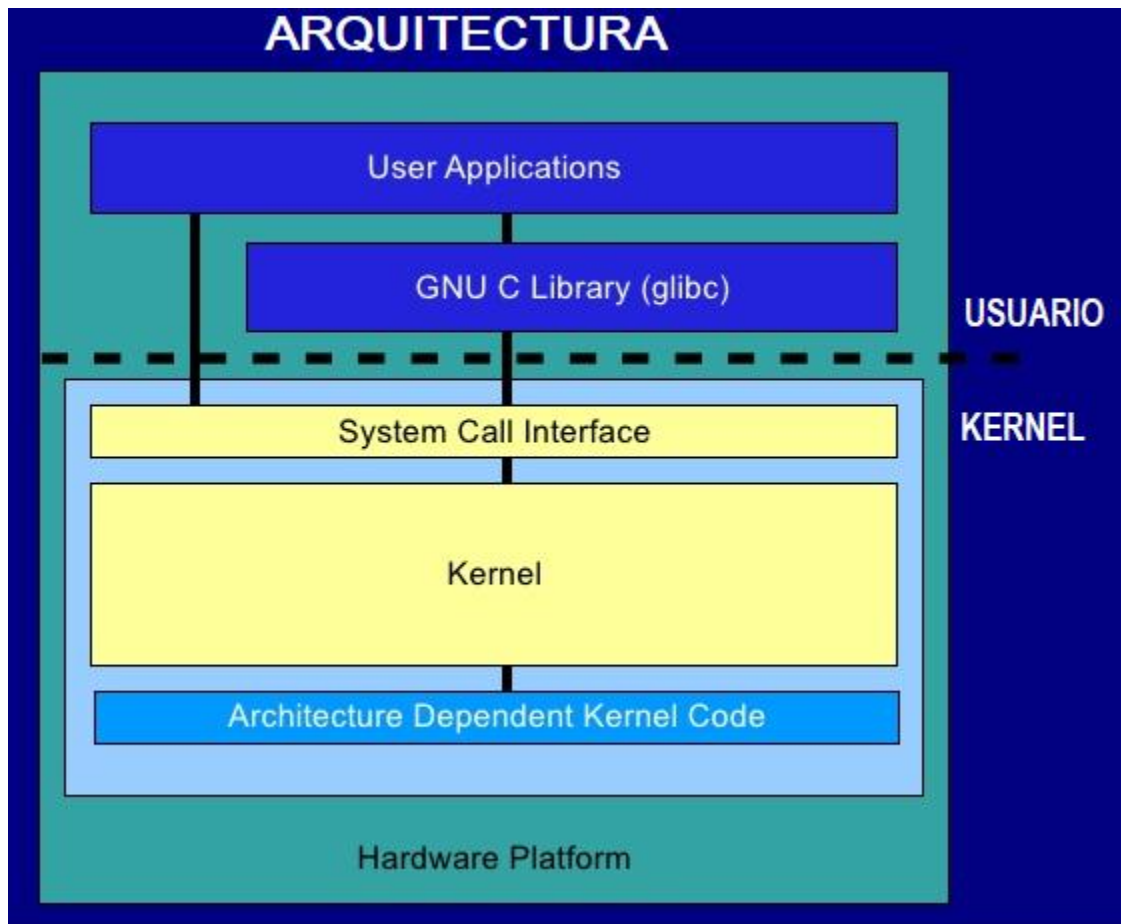
Mejor característica	Actualización del kernel sin reiniciar	Feta ZFS: sistema de archivos de próxima generación DTrace: seguimiento dinámico del kernel
Versiones	Las diferentes versiones de Linux son Redhat, Ubuntu, OpenSuse, etc.	Las diferentes versiones de Unix son HP-UX, AIS, BSD, etc.
Tipo de archivo admitido	Los sistemas de archivos compatibles con el tipo de archivo como xfs, nfs, cramfs ext 1 a 4, ufs, devpts, NTFS.	Los sistemas de archivos compatibles con los tipos de archivos son zfs, hfx, GPS, xfs, vxfs.
Portabilidad	Linux es portátil y se inicia desde una memoria USB	Unix no es portátil
Código fuente	La fuente está disponible para el público en general.	El código fuente no está disponible para nadie.

BLOQUE II

TEMA 04

2.2 ARQUITECTURA UNIX/LINUX

La arquitectura de Unix está basada en capas o niveles. El traspaso de información entre capas se hace a través de **cepos o traps**.



Capa Hardware \equiv Todos los dispositivos físicos conectados que se deben manejar por el SO.

Kernel \equiv Por encima de la capa hardware, e interactuando directamente con ella, se sitúa el Kernel, encargado de la gestión de procesos, sistema de archivos, E/S, etc.

A los procesos que trabajan a este nivel se les llama procesos en modo kernel. La comunicación con las capas superiores se realiza mediante la interface **de llamadas del sistema**.

Biblioteca estándar \equiv Se ubica por encima del kernel. Incluye todas las funciones estándar del lenguaje C. A este nivel se trabaja en modo usuario.

Todos los procesos comienzan como modo de usuario. El kernel no es un proceso, es un controlador de procesos.

El paso de modo usuario a kernel o viceversa se realiza a través de cepos o traps que crean una interrupción para acceder a la interface de llamadas al sistema y al resto de los componentes de nivel kernel. El paso del modo kernel a usuario es un retorno tras la realización de la petición que motivó el paso al modo kernel.

BLOQUE II

TEMA 04

Aplicaciones de Usuario ≡ Incluye el Shell (intérprete de comandos) y los programas de utilidad que se ejecutan a través del Shell (compilador, navegador web, reproductor multimedia, editor de texto, etc.)

2.3 ARRANQUE DEL SISTEMA OPERATIVO

2.3.1 NIVELES DE EJECUCIÓN

Un primer punto importante en el análisis del comportamiento local del sistema, es su funcionamiento en los llamados **niveles de ejecución (o runlevels)**, que determinan (en el nivel) el modo actual de trabajo del sistema, y los servicios que se proporcionan.

Un nivel de ejecución es básicamente una configuración de programas y servicios que se ejecutarán orientados a un determinado funcionamiento.

Cada uno de estos modos, o niveles de ejecución, tiene su propia lista de procesos y servicios que están activados o desactivados. Desde el momento en que Linux arranca, siempre está en algún nivel de ejecución.

Hay siete niveles de ejecución diferentes en Linux, numerados del cero al seis.

- ❖ Runlevel 0 → Apaga el sistema.
- ❖ Runlevel 1 → Modo monousuario, que se utiliza para medidas de mantenimiento o administrativas. Este modo también se conoce como nivel de ejecución S (la S significa un solo usuario).
- ❖ Runlevel 2 → Modo multiusuario. Este nivel de ejecución no utiliza ningún servicio de red.
- ❖ Runlevel 3 → Modo multiusuario con interconexión. Este es el nivel de ejecución normal si se utiliza un sistema que NO arranca en una GUI (interfaz gráfica de usuario).
- ❖ Runlevel 4 → No se utiliza. El usuario puede personalizar este nivel de ejecución para sus propios fines.
- ❖ Runlevel 5 → Igual que el nivel de ejecución 3, pero también inicia un gestor de pantalla. Este es el nivel de ejecución normal SI arranca un sistema que se inicia en una GUI.
- ❖ Runlevel 6 → Para todos los programas y servicios. Reinicia el sistema.

2.3.2 GESTORES DE ARRANQUE

Un cargador de arranque (**boot loader o boot Manager**), es un pequeño programa almacenado en la tabla de particiones MBR o GUID que ayuda a cargar el sistema operativo en la memoria. Sin el cargador de arranque el sistema no se cargará en la memoria.

El **espacio de usuario temprano** se utiliza en las versiones más recientes del núcleo Linux para sustituir tantas funciones como sea posible que originalmente se harían en el núcleo durante el proceso de inicio. Los usos típicos del espacio de usuario temprano son para detectar que

BLOQUE II**TEMA 04**

controladores de dispositivos (Drivers) son necesarios para cargar el sistema de archivos del espacio de usuario principal y cargarlos desde un sistema de archivos temporal.

FIRMWARE: BIOS y EFI (UEFI)

Antes de hablar de los cargadores de arranque es necesario hablar de dos de los firmwares más usados en equipos x86-64, estos son BIOS (Basic Input/Output System) y EFI (Extensible Firmware Interface) o su versión posterior UEFI (Unified EFI).

De una u otra forma, el firmware de su ordenador lee el cargador de arranque en memoria desde el disco duro y lo ejecuta. Por su parte, el cargador de arranque se encarga de cargar el kernel de Linux en memoria y lo empieza a ejecutar, por tanto, la configuración del cargador de arranque en el disco es clave para que el firmware pueda localizarlo y comenzar con el proceso de inicio del sistema.

A la secuencia completa de arranque de un sistema (es decir la delegación de BIOS en el cargador de arranque y de este en el kernel del OS) se le conoce como Bootstrapping o Bootstrap Loader.

Tabla de los 2 principales firmwares y sus cargadores de arranque:

FIRMWARE	
<i>BIOS</i>	<i>UEFI</i>
<i>Tabla de particiones</i>	<i>Tabla de particiones</i>
<ul style="list-style-type: none"> • MBR (Master Boot Record) 	<ul style="list-style-type: none"> • GPT (GUID Partition Table) • MBR (Master Boot Record)
Gestores de arranque	Gestores de arranque
<ul style="list-style-type: none"> • GRUB Legacy • GRUB • LILO • NEOGRUB • SYSLINUX 	<ul style="list-style-type: none"> • GRUB • SYSLINUX • EFISTUB • GUMMIBOOT • rEFInd • ELILO

BLOQUE II

TEMA 04

GNU GRUB y GRUB 2 \equiv GNU GRand Unified Bootloader. Es un cargador de arranque múltiple, desarrollado por el proyecto GNU que nos permite elegir qué Sistema Operativo arrancar de los instalados.

Las características principales son:

- Reconoce múltiples formatos ejecutables.
- Soporta kernels que no cumplen especificación multiboot.
- Admite comandos de configuración y puede cargar una configuración preestablecida.
- Provee interfaz de menú.
- Tiene una interfaz de línea de comandos flexible.
- Admite muchos sistemas de ficheros (BtrFS, ext2/ext3/ext4, FAT12/FAT16/FAT32, exFAT, HFS, HFS+ ISO9660, nilfs2, NTFS, ReiserFS, ZFS, ROMFS, JFS, XFS, etc).
- Acceso a datos ubicados en cualquier dispositivo instalado.
- Independiente de la geometría de la unidad.
- Detecta toda la RAM instalada.
- Admite direccionamiento de bloque lógico (LBA).
- Arranque de red.
- Terminales remotos para permitir el control desde una estación remota.

GRUB2 ha sustituido al original GRUB que pasó a llamarse desde entonces GRUB Legacy.

Las diferencias más importantes entre GRUB 2 y GRUB Legacy son:

- El fichero de configuración de GRUB 2 es 'grub.cfg' en lugar del 'grub.conf' o 'menu.lst'. Se añade nueva sintaxis y muchos nuevos comandos.
- Los números de partición en los nombres de los dispositivos GRUB ahora empiezan en 1, en lugar de en 0.
- GRUB 2 tiene formas más confiables de encontrar sus propios ficheros y los de los kernels de destino en sistemas de varios discos, y tiene comandos para buscar dispositivos usando etiquetas del sistema de archivos o identificadores únicos universales (UUID).
- GRUB 2 soporta muchos más sistemas de ficheros, incluyendo pero no limitado a ext4, HFS+, and NTFS.
- GRUB 2 puede leer ficheros directamente de dispositivos LVM y RAID.
- GRUB 2 tiene disponible un terminal gráfico y un sistema de menú gráfico
- GRUB 2 introduce muchas instalaciones en módulos cargados dinámicamente, lo que permite que la imagen del núcleo sea más pequeña y que se construya de formas más flexibles

BLOQUE II

TEMA 04

LILO ≡ Linux LOader. Cada vez más en desuso. Sus características básicas son las siguientes:

- Permite el arranque de varios sistemas operativos
- Permite arrancar desde discos duros y flexibles
- Se instala el MBR
- Al iniciar el sistema LILO solamente puede acceder a los drivers de la BIOS para acceder al disco duro. Por esta razón en BIOS antiguas el área de acceso está limitado a los cilindros numerados de 0 a 1023 de los dos primeros discos duros. En BIOS posteriores LILO puede utilizar sistemas de acceso de 32 bits permitiéndole acceder a toda el área del disco duro.

En general, LILO funciona de forma parecida a GRUB a excepción de tres diferencias:

- No posee ninguna interfaz del comando interactiva.
- Almacena información sobre la localización del kernel o de si otro sistema operativo se debe cargar en el MBR.
- No puede leer las particiones ext2.

BURG ≡ Burg es un gestor de arranque basado en Grub2. Se utiliza un formato nuevo objeto que le permite ser construido en una gama más amplia de sistema operativo, incluyendo GNU/Linux, Windows, OSX, Solaris, FreeBSD, etc. También tiene un sistema de menús muy configurable que funciona tanto en modo texto y gráfico.

La idea de BURG es proveer un cargador de arranque con aspecto visual, capaz de mostrar fondos de escritorio, iconos, y animaciones en lugar de solo texto como GRUB.

Características

- La principal característica de BURG es que mejora totalmente el aspecto visual, asemejándolo con el cargador de arranque de las computadoras Macintosh.
- BURG posee un alto índice de portabilidad permitiendo instalarlo en diversas unidades de disco, unidades de estado sólido o memorias USB
- Programación totalmente idéntica a GRUB pero reescrita desde 0, imitando sus funciones.
- BURG posee además una aplicación para mostrar una vista previa sin tener que reiniciar la computadora.

SYSLINUX ≡ Syslinux es un surtido de cargadores de arranque muy ligeros, que permiten el arranque desde el CD-ROM, desde una red, etc. Soporta sistemas de archivos como FAT para MS-DOS, y ext2, ext3, ext4 para Linux. También admite Btrfs de un solo dispositivo sin comprimir.

Un detalle a tener en cuenta es que Syslinux sólo accede a los archivos de su propia partición, por tanto, no tiene la capacidad de inicio de varios sistemas de archivos.

El proyecto SYSLINUX abarca un conjunto de gestores de arranque ligeros, es decir, diseñados para poder arrancar un sistema operativo Linux desde una memoria USB, un disco flexible, CD-ROM, etc.

BLOQUE II

TEMA 04

2.4 TIPOS DE PARTICIONES

Al instalar una distribución de Linux, esta va a necesitar 3 particiones diferentes (aunque pueden ser solo 2).

- Partición primaria para el directorio raíz: /
- Partición lógica para el área de intercambio: swap
- Partición lógica para los archivos, el directorio: /home

La partición para el directorio home no es de uso obligatorio, se puede crear una partición para la raíz y otra para el área de intercambio, y los archivos se almacenarán en el directorio raíz sin problemas.

La partición para el área de intercambio es utilizada como un respaldo de la memoria RAM, para almacenamiento temporal (memoria virtual). El tamaño para el área de intercambio se recomienda sea el doble de la memoria RAM que tenga el ordenador.

Al definir la partición también se debe definir un sistema de archivos (vistos en el tema 2) y un punto de montaje. De la siguiente manera:

- Directorio raíz: Sistema de ficheros ext4 transaccional. Punto de montaje: /
- Directorio home: Sistema de ficheros ext4 transaccional. Punto de montaje: /home
- Área de intercambio: utilizar como área de intercambio. No tiene formato.

La memoria Swap se puede implementar de dos formas:

- Mediante una partición
- Mediante un fichero.

En caso de querer habilitar el uso de Swap mediante un fichero, lo que debemos hacer es ejecutar los siguientes comandos:

- Crear el fichero de intercambio (Ejemplo 1Gb): `sudo fallocate -l 1G /swapfile`
- Permisos para que solo root pueda escribir en dicho fichero: `sudo chmod 600 /swapfile`
- Dar al archivo una estructura para poder funcionar como archivo de intercambio:

`sudo mkswap /swapfile`

- Activarlo: `sudo swapon /swapfile`

BLOQUE II

TEMA 04

Relacionado con la memoria Swap, tenemos el concepto de **Swappiness**.

Swappiness → Elegir cuándo queremos que Linux use el Swap. Por defecto, muchas distribuciones de Linux tienen configurado un swappiness por defecto de 60. Esto significa que el Swap no se usa hasta que se use el 60% de la memoria RAM. A partir de ese umbral es cuando se empieza a usar la memoria de intercambio.

Este valor podemos modificarlo editando el siguiente fichero con un editor con permisos de root:
`cat /proc/sys/vm/swappiness`

MBR Y GPT

Hay dos formas principales de almacenar información de partición en discos duros. El primero es MBR (Master Boot Record) y el segundo es GPT (GUID Partition Table).

MBR ≡ Este es un remanente de los primeros días de MS-DOS y durante décadas fue el esquema de particionamiento estándar en los PC. La tabla de particiones se almacena en el primer sector del disco, llamado Boot Sector, junto con un cargador de arranque, que en los sistemas Linux suele ser el GRUB. Pero MBR tiene una serie de limitaciones que dificultan su uso en sistemas modernos, como la imposibilidad de utilizar discos de más de 2 TB de tamaño y el límite de solo 4 particiones primarias por disco.

GPT ≡ Un sistema de particiones que aborda muchas de las limitaciones de MBR. No existe un límite práctico en el tamaño del disco, y el número máximo de particiones está limitado solo por el propio sistema operativo. Se encuentra más comúnmente en máquinas más modernas que **usan UEFI** en lugar del antiguo BIOS de PC.

Durante las tareas de administración del sistema es muy posible que encuentre ambos esquemas en uso, por lo que es importante saber cómo usar las herramientas asociadas a cada uno para crear, eliminar o modificar particiones.

- **FDISK** ≡ Gestiona particiones MBR
- **GDISK** ≡ Gestiona particiones GPT
- **GNU Parted** ≡ Es un editor de particiones muy poderoso (de ahí el nombre) que se puede usar para crear, eliminar, mover, redimensionar, rescatar y copiar particiones. Puede funcionar con discos GPT y MBR

Cualquier de las tres herramientas anteriores permiten crear particiones de intercambio (swap), es decir, que la partición de intercambio puede ser tanto de tipo GPT como MBR.

BLOQUE II

TEMA 04

2.5 SISTEMA DE FICHEROS

En Linux existen básicamente 5 tipos de archivos:

Archivos ordinarios \equiv Contienen la información con la que trabaja cada usuario.

Enlaces físicos o duros (hard links) \equiv No es específicamente una clase de archivo sino un segundo nombre que se le da a un archivo. En el caso que dos usuarios necesiten compartir información de un mismo archivo, si cada uno tuviera una copia del archivo se soluciona el problema, pero las modificaciones que realice un usuario no las vería el otro. Sin embargo, si se crea un enlace duro al archivo para cada usuario cada vez que uno de ellos modifique cualquier cosa en el archivo, el otro lo podrá ver puesto que realmente están viendo y modificando el mismo archivo.

Enlaces simbólicos \equiv También se utilizan para asignar un segundo nombre a un archivo. La diferencia con los enlaces duros es que los simbólicos solamente hacen referencia al nombre del archivo original, mientras que los duros hacen referencia al i-nodo en el que están situados los datos del archivo original. De esta manera, si tenemos un enlace simbólico y borramos el archivo original perderemos los datos, mientras que si tenemos un enlace duro los datos no se borrarán hasta que se hayan borrado todos y cada uno de los enlaces duros que existen hacia esos datos en el sistema de ficheros.

Directorios \equiv Son archivos especiales que contienen referencias a otros archivos o directorios. Un directorio no es más que un fichero que contiene los nombres de los ficheros (o directorios) que contiene junto con el número de i-nodo que contiene la información de cada uno de ellos.

Archivos especiales \equiv Suelen representar dispositivos físicos, como unidades de almacenamiento, impresoras, terminales, etc. En Linux, todo dispositivo físico que se conecte al ordenador está asociado a un archivo. Linux trata los archivos especiales como archivos ordinarios

El sistema de ficheros de Linux permite al usuario crear, borrar y acceder a los ficheros sin necesidad de saber el lugar exacto en que se encuentran. En Linux no existen unidades físicas, sino ficheros que hacen referencia a ellas, integrados en la estructura de ficheros como cualquier otro.

El sistema de ficheros de Linux consta de tres partes importantes:

- Súper bloque
- Tabla de inodos
- Bloques de datos



El **bloque de carga o bloque cero** de cada sistema está reservado para almacenar un programa que utiliza el sistema para gestionar el resto de las partes del sistema de ficheros.

El **superbloque o bloque uno** contiene la información sobre el sistema de ficheros.

BLOQUE II

TEMA 04

La **tabla de i-nodos** es el equivalente a las entradas de la FAT. Por cada fichero, Linux tiene asociado un elemento en esta tabla identificado por un **número entero**. Este número identifica la ubicación del archivo dentro del área de datos.

El **área de datos** ocupa el resto del disco y es equivalente a la zona de datos en FAT. En esta zona, como su nombre indica, están almacenados los ficheros y directorios de nuestro sistema.

Cada i-nodo contiene información de un fichero o directorio. Cada archivo tiene un i-nodo único.

En concreto, en un i-nodo se guarda la siguiente información:

- El identificador de dispositivo del dispositivo que alberga al sistema de archivos.
- El número de i-nodo que identifica al archivo dentro del sistema de archivos
- La longitud del archivo en bytes.
- El identificador de usuario del creador o un propietario del archivo con derechos diferenciados
- El identificador de grupo de un grupo de usuarios con derechos diferenciados
- El modo de acceso: capacidad de leer, escribir, y ejecutar el archivo por parte del propietario, del grupo y de otros usuarios.
- Las marcas de tiempo con las fechas de última modificación (mtime), acceso (atime) y de alteración del propio i-nodo (ctime).
- El número de enlaces, esto es, el número de nombres (entradas de directorio) asociados con este i-nodo. El número de enlaces se emplea por el sistema operativo para eliminar el archivo del sistema de ficheros, tanto el i-nodo como el contenido, cuando se han borrado todos los enlaces y el contador queda a cero.
- La estructura de punteros, para direccionar hacia los bloques de datos (contenido) del archivo. Está compuesta por:
 - Doce punteros que apuntan directamente a bloques de datos del archivo (punteros directos)
 - Un puntero de indirección simple (apunta a un bloque de punteros, los cuales apuntan a bloques de datos del archivo)
 - Un puntero de indirección doble (apunta a un bloque de punteros, los cuales apuntan a otros bloques de punteros, estos últimos apuntan a bloques de datos del archivo)
 - Un puntero de indirección triple (apunta a un bloque de punteros que apuntan a otros bloques de punteros que apuntan a otros bloques de punteros que luego apuntan a bloques de datos del archivo)

BLOQUE II**TEMA 04****2.6 ESTRUCTURA DE DIRECTORIOS**

Linux tiene una estructura de directorios (o carpetas) bien definida, donde cada directorio tiene una finalidad muy concreta, y en él se guardan los archivos correspondientes.

La ruta raíz (la primera carpeta del sistema) en Linux es /. Desde esa ruta / nos podemos encontrar varias carpetas, cada una con una finalidad concreta:

/	Carpeta raíz del sistema. Equivalente a C:\ en Windows.
/bin	ficheros binarios de comandos que requiere el sistema operativo para funcionar
/boot	Archivos de arranque. Equivale a C:\bootmgr en Windows.
/dev	Dispositivos físicos de hardware.
/etc	Archivos de configuración. Equivale al registro de Windows .
/home	Carpetas del usuario. Equivale a C:\Users en Windows.
/lib	Bibliotecas del sistema. Equivale a C:\Windows\System32 .
/lost+found	Archivos corruptos o recuperados.
/media	Medios extraíbles. Generalmente, carpetas de CD/DVD, usb, unidades extraíbles...
/mnt	Montaje temporal reservado para el administrador de sistemas.
/opt	Aplicaciones o paquetes adicionales (<i>opcionales</i>).
/proc	Procesos internos del sistema.
/root	Carpeta personal del administrador.
/run	Información reciente.
/sbin	Archivos binarios reservados para root. Sólo accesible si tienes privilegios.
/srv	Datos específicos para servicios.
/sys	Nueva versión o evolución de /proc/
/tmp	Archivos temporales. Muchas veces, eliminados al reiniciar el sistema.
/usr	Datos de paquetes instalados. Equivale a C:\Archivos de Programa en Windows.
/var	información de datos temporales o variables con el objetivo de alertar y erradicar problemas detectados: Logs, cache, variables, etc...

BLOQUE II

TEMA 04

2.7 ARCHIVOS IMPORTANTES EN LINUX

Administrar usuarios y grupos en un equipo con Linux es uno de los aspectos clave de la administración del sistema. De hecho, Linux es un sistema operativo multiusuario en el que varios de estos pueden usar la misma máquina al mismo tiempo.

La información sobre usuarios y grupos se almacena en cuatro archivos dentro del árbol de directorios `/etc/`:

`/etc/passwd` ≡ Es un archivo de siete campos delimitados por dos puntos que contiene información básica sobre los usuarios.

- Username → El nombre utilizado cuando el usuario inicia sesión en el sistema.
- Password → La contraseña cifrada (o una x si se usan contraseñas ocultas).
- User ID (UID) → El número de ID asignado al usuario en el sistema.
- Group ID (GID) → El número de grupo primario del usuario en el sistema.
- GECOS → Un campo de comentario opcional, que se utiliza para agregar información adicional sobre el usuario (como el nombre completo). El campo puede contener múltiples entradas separadas por comas.
- Home directory → La ruta absoluta del directorio de inicio del usuario.
- Shell → La ruta absoluta del programa que se inicia automáticamente cuando el usuario inicia sesión en el sistema (generalmente un Shell interactivo como `/bin/bash`).

`/etc/group` ≡ Es un archivo de cuatro campos delimitados por dos puntos que contienen información básica sobre grupos.

- Group Name → El nombre del grupo.
- Group Password → La contraseña cifrada del grupo (o una x si se usan contraseñas ocultas).
- Group ID (GID) → El número de identificación asignado al grupo en el sistema.
- Member list → Una lista delimitada por comas de los usuarios que pertenecen al grupo, excepto aquellos para quienes este es el grupo primario.

`/etc/shadow` ≡ Es un archivo legible solo por usuarios root y usuarios con privilegios de root. Además, contiene las contraseñas cifradas de los usuarios separadas en distintas líneas. Cada línea consta de nueve campos delimitados por dos puntos:

- Username → El nombre utilizado cuando el usuario inicia sesión en el sistema.
- Encrypted password → La contraseña cifrada del usuario (si el valor es !, La cuenta está bloqueada).
- Date of last password change → La fecha del último cambio de contraseña, como número de días desde 01/01/1970. Un valor de 0 significa que el usuario debe cambiar la contraseña en el siguiente acceso.
- Minimum password age → El número mínimo de días después de un cambio de contraseña que debe pasar antes de que el usuario pueda cambiarla nuevamente.

BLOQUE II

TEMA 04

- Maximum password age → El número máximo de días que deben transcurrir antes de que se requiera un cambio de contraseña.
- Password warning period → El número de días antes de que caduque la contraseña, durante los cuales se advierte al usuario que se debe cambiarla.
- Password inactivity period → El número de días después de que caduca una contraseña durante el cual el usuario debe actualizarla. Después de este período, si el usuario no cambia la contraseña, la cuenta se deshabilitará.
- Account expiration date → La fecha, como número de días desde el 01/01/1970 en que se deshabilitará la cuenta de usuario. Un campo vacío significa que la cuenta de usuario nunca caducará.
- A reserved field → Un campo reservado para uso futuro.

/etc/gshadow ≡ Es un archivo legible solo por root y por usuarios con privilegios de root que contiene contraseñas cifradas para grupos, cada uno en una línea separada. Cada línea constante de cuatro campos delimitados por dos puntos:

- Group name → El nombre del grupo.
- Encrypted password → La contraseña cifrada para el grupo (se usa cuando un usuario que no es miembro del grupo, desea unirse al grupo usando el comando newgrp, si la contraseña comienza con !. Nadie puede acceder al grupo con newgrp).
- Group administrators → Una lista delimitada por comas de los administradores del grupo (pueden cambiar la contraseña del grupo y pueden agregar o eliminar miembros del grupo con el comando gpasswd).
- Group members → Una lista delimitada por comas de los miembros del grupo.

Todos estos archivos se actualizan mediante un conjunto de herramientas de línea de comandos para la administración de usuarios y grupos, descritas en capítulos posteriores.

/etc/skel ≡ (Directorio). Cuando se agrega una nueva cuenta de usuario, incluso al crear su directorio de inicio, el directorio de inicio recién creado se llena con archivos y carpetas que se copian del directorio de skel (por defecto /etc/skel). La idea detrás de esto, es simple: un administrador del sistema quiere agregar nuevos usuarios que tengan los mismos archivos y directorios en su hogar. Por lo tanto, si desea personalizar los archivos y carpetas que se crean automáticamente en el directorio de inicio de las nuevas cuentas de usuario, debe agregar estos nuevos archivos y carpetas al directorio de skel.

BLOQUE II**TEMA 04****2.8 PERMISOS EN LINUX**

Al ser un sistema multiusuario, Linux necesita alguna forma de rastrear quién es dueño de cada archivo y si un usuario puede o no realizar acciones en ese en el mismo. Lo anterior es para garantizar la privacidad de los usuarios que deseen mantener la confidencialidad del contenido de sus ficheros, así como para garantizar la colaboración al hacer que ciertos archivos sean accesibles para múltiples usuarios.

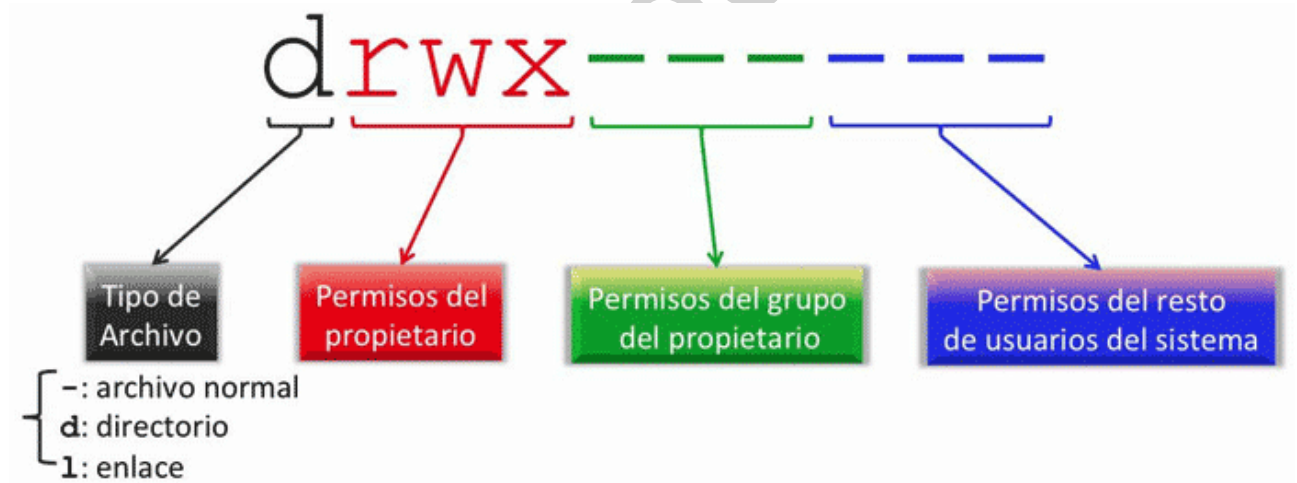
Esto se hace a través de un sistema de permisos de tres niveles: cada archivo en el disco es propiedad de un usuario y un grupo de usuarios y tiene tres conjuntos de permisos: uno para su propietario, otro para el grupo propietario del archivo y otro para todos los demás.

Al visualizar un listado de archivos en Linux (por ejemplo, con el comando *ls*), podemos obtener una información similar a la siguiente:

drwxrwxrwx 2 carol 4096 Dec 10 15:57 Another_Directory

La primera columna, muestra el tipo de archivo y los permisos.

- El primer carácter identifica el tipo de archivo
- Los tres siguientes identifican los permisos del usuario propietario
- Los tres siguiente identifican los permisos del grupo propietario
- Los tres siguientes identificas los permisos para todos los usuarios



Estos permisos pueden ser cambiados con comandos como *chmod*, que se verán más adelante.

BLOQUE II

TEMA 04

Tipo de Archivo

Los tres tipos de archivos más comunes son:

- **- (guion. archivo normal)** Un archivo puede contener datos de cualquier tipo. Los archivos se pueden modificar, mover, copiar y eliminar.
- **d (directorio)** Un directorio contiene otros archivos o directorios y ayuda a organizar el sistema de archivos. Técnicamente, los directorios son un tipo especial de archivo.
- **l (Letra ele. enlace suave)** ≡ Este "archivo" es un puntero a otro archivo o directorio en otra parte del sistema de archivos.

Además de esos, hay otros tres tipos de archivos:

- **b (dispositivo de bloque)** ≡ Este archivo representa un dispositivo virtual o físico, generalmente discos u otros tipos de dispositivos de almacenamiento. Por ejemplo, el primer disco duro del sistema podría estar representado por /dev/sda.
- **c (dispositivo de caracteres)** ≡ Este archivo representa un dispositivo virtual o físico. Los terminales (como el terminal principal en /dev/ttyS0) y los puertos seriales son ejemplos comunes de dispositivos de caracteres.
- **s (socket)** ≡ Los sockets sirven como "conductos" para pasar información entre dos programas.

Comprendiendo los permisos

Permisos de archivo ≡ se muestran justo después del tipo de archivo, como tres grupos de tres caracteres cada uno, en el orden r, w y x. Un guion representa la falta de un permiso particular.

r → read (leer). Significa permiso para abrir un archivo y leer su contenido.

w → write (escribir). Significa permiso para modificar o eliminar un archivo.

x → execute (ejecutar). Significa que el archivo se puede ejecutar como un script.

Ejemplos:

- Un archivo con permisos **rw-** se puede leer y escribir, pero no se puede ejecutar.
- Un archivo con permisos **r--** solo se puede leer, ni modificar ni ejecutar.

BLOQUE II

TEMA 04

Permisos en directorios ≡

r → read. Significa permiso para leer el contenido del directorio, como los nombres de archivo, pero no implica permiso para leer los archivos en sí mismos.

w → write. Significa permiso para crear o eliminar archivos en un directorio o cambiar sus nombres, permisos y propietarios. Si un usuario tiene permiso de escritura en un directorio, el usuario puede cambiar los permisos de cualquier archivo en el directorio, incluso si el usuario no tiene permisos en el archivo o si el archivo es propiedad de otro usuario.

x → execute. Significa permiso para ingresar a un directorio, pero no para enumerar sus archivos (para eso se necesita el permiso r).

Permisos especiales

Además de los permisos de lectura, escritura y ejecución para usuarios, grupos y otros, cada archivo puede tener otros tres permisos especiales que pueden alterar la forma en que funciona un directorio o cómo se ejecuta un programa. Son los siguientes:

Sticky Bit (bit de permanencia) ≡ El sticky bit, también llamado restricted deletion flag, en modo simbólico está representado por una t dentro de los permisos del otro. Esto se aplica solo a los directorios, y en Linux evita que los usuarios eliminen o cambien el nombre de un archivo en un directorio a menos que sean dueños de ese archivo o directorio.

Set GID ≡ Establecer GID, también conocido como SGID o "Set Group ID bit", en modo simbólico está representado por una s en los permisos de grupo. Esto se puede aplicar a archivos ejecutables o directorios. En los archivos ejecutables, otorgará la ejecución del archivo con privilegios del grupo. Cuando se aplica a los directorios, hará que cada archivo o directorio creado debajo herede el grupo del directorio principal.

Set UID ≡ SUID, también conocido como "Set User ID", está representado por una s en los permisos user en modo simbólico. Solo se aplica a los archivos y su comportamiento es similar al bit SGID, pero el proceso se ejecutará con los privilegios del usuario que posee el archivo. Los archivos con el bit SUID muestran una s que reemplaza la x en los permisos para el usuario.

BLOQUE II**TEMA 04****2.9 DISPOSITIVOS**

Para Linux todo es un archivo, incluyendo dispositivos como discos duros, cdroms, disquetes, unidades de cinta, memorias usb, etc, así como dispositivos de comunicación como puertos serie y paralelos, modem, etc, incluso también las consolas o terminales son dispositivos asociados a un archivo.

Estos dispositivos son enlazados (linked) a un dispositivo de archivo, es decir un dispositivo físico es representado o asociado a un archivo. Estos archivos se encuentran bajo el directorio **/dev**.

Existen tres tipos especiales de dispositivo:

- Dispositivos de caracteres: les llega el flujo de datos de caracter en caracter. (ttys)
- Dispositivos de bloque: les llega el flujo de datos en bloques. Soportan acceso aleatorio a los datos. (hds)
- Pseudo-dispositivos: no corresponden a ningún hardware específico

Los dispositivos en Linux son identificados con un identificador de dos o tres letras, además si el dispositivo admite particiones se utiliza una progresión numérica o alfabética para identificar la partición.

En la siguiente tabla se indica el identificador de tipo dispositivo y su descripción:

Tipo	Dispositivo
hd	Discos duros IDE y dispositivos IDE como cdroms
sd	Discos duros SATA / SCSI
scd	Cdrom SCSI
st	Unidades de cinta SCSI
ht	Unidades de cinta IDE
fd	Unidades de disquetes
lp	Puertos paralelos
tty	Terminales o consolas
pty	Terminales remotas o de red, incluyendo las abiertas en Window X
ttyS	Puertos seriales
ttyUSB	conversor serie-usb
cua	Puertos de comunicación
eth	Tarjetas o interfaces de red ethernet

BLOQUE II

TEMA 04

En los dispositivos que aceptan particiones, el nombramiento de los mismos está dividido en tres partes:

1. Tipo de disco (sd \equiv SCSI/SATA, hd \equiv IDE)
2. Identifica el número de disco (a, b, c, d)
3. Identifica el número de partición

Ejemplos:

- /dev/sda1 \rightarrow sd (disco SATA), a (primer disco), 1 (Primera partición del disco)
- /dev/hdb1 \rightarrow hd (disco IDE), b (segundo disco, esclavo del primero), 1 (Primera partición del disco).

Existen otros dispositivos especiales como:

- /dev/null: Acepta todo tipo de datos de entrada y los descarta. No produce ninguna salida
- /dev/full: Dispositivo que siempre está lleno
- /dev/loop: Dispositivo de bucle
- /dev/zero: Produce un flujo de datos continuo de caracteres NULL (cero)
- /dev/random: Produce un flujo de datos de números aleatorios. Bloquea
- /dev/urandom: Produce un flujo de datos de números aleatorios. No bloquea.

2.10 PROCESOS

Un proceso es nada más que un programa en ejecución. Para ser más preciso digamos que es, el estado actual de ejecución en un dado momento. Este consiste del código del programa, los recursos usados por el programa y la información utilizada para mantener el estado actual del programa. Todo Programa en ejecución tiene su propio proceso. (Algunos programas tienen más de un proceso) El kernel utiliza la información del estado del proceso para asignarle recursos, como lo es por ejemplo tiempo del CPU, memoria, archivos y E/S además de otros recursos. También utiliza esta información para determinar cómo cambiar a otro proceso.

A veces los procesos son referidos como tarea (task). GNU/Linux es un sistema operativo multitarea, lo cual da la impresión o ilusión de que muchos procesos se ejecutan simultáneos. El kernel se encarga de que los procesos se dividan el tiempo del CPU en ciclos y automáticamente cambia de un proceso a otro. Este cambio es denominado "**Cambio de Contexto**". Linux implementa "multitarea con derecho preferente", lo que simplemente significa que el kernel no necesita ninguna colaboración del programa para llevarlo a cabo, si no que el simplemente fuerza un cambio de contexto cuando lo considere necesario.

La información del estado del proceso es almacenada en Bloques de Control de Procesos (Process Control Blocks PCB), supervisados por el Kernel. A cada proceso es asignado un identificador de proceso (Process ID - PID) el cual es utilizado para localizarlo dentro de la tabla de los procesos del kernel.

Todos los procesos, excepto el init, tienen un proceso padre (quien lo genero). El init al ser generado por el kernel al arranque no tiene proceso padre.

BLOQUE II

TEMA 04

Hay mucha información almacenada en el PCB de cada proceso. La mayoría de esta información es solo relevante e importante para el kernel, mientras que existe otras informaciones que si son importantes para todo el sistema. Algunos de los ítems en el PCB más importantes:

- ID del Proceso (PID)
- Parent Process ID (PPID)
- Verdadero User y Group IDs (UIDs y GIDs)
- Efectivo User y Group IDs (EUIDs y EGIDs)
- Process State (Estado del Proceso)
- Signal State (Estado de la Señal)

El PID es un entero único que se usa para no perder de vista de los procesos. Los PIDs son asignados en orden numérica. Es por eso que el init, el primer proceso que se ejecuta en el sistema, es siempre el proceso número 1. El PID más grande en Linux es el 32.767 (entero más grande que puede ser representado por 2 bytes de memoria). El proceso creado después del 32,767 obtendría el entero número 1, pero en razón de no poder repetir asignaciones para que sean únicas, un nuevo proceso no puede tener este número puesto que ya está en uso por el init. Por esto el kernel elija el próximo número que no sea el PID de un proceso que se está ejecutando.

El PPID es el PID del proceso padre.

ESTADOS DE LOS PROCESOS

- R Running → Ejecutándose
- S Sleeping → Durmiendo
- T Terminating → Terminando
- D Device I/O → Dispositivo E/S
- Z Zombie → Zombis

Los procesos en el estado de ejecutándose o se encuentran en ejecución en el CPU o están a punto de iniciar a correr en el CPU desde que exista la disponibilidad de acceso.

Durmiendo es el estado en cual un proceso cae cuando está en espera de que un evento suceda para el despertar y entonces continuar procesando. Casi siempre espera por un tipo de operación de E/S.

Un proceso terminando es un proceso que ha sido suspendido por el usuario.

Un proceso en el estado D (Dispositivo E/S) no puede ser interrumpido; el kernel está ocupado gestionando algún tipo de proceso de Entrada/Salida.

Procesos Zombis. Cuando un proceso hijo termina antes que el proceso padre, el kernel mantiene información acerca del proceso hijo para mantener al proceso padre informado de cosas como el estatus de salida. El proceso padre es notificado que el proceso ha terminado. El proceso padre debe indicarle al kernel que ha terminado la necesidad del acceso a la información que el kernel mantiene sobre el proceso hijo para

BLOQUE II

TEMA 04

que el kernel pueda librar la memoria asociada con este proceso hijo. El proceso padre provee esta información o indicación utilizando el sistema de señales de espera (wait system calls). Hasta que el proceso padre no indica estas llamadas, un proceso hijo terminado no puede ser removido del sistema y entra en el estado Zombie; el proceso está muerto, pero no puede ser destruido.

Los procesos Zombis no ocupan tiempo del CPU, pero ocupan un espacio en la tabla de los procesos y es reducido del límite de procesos colocados sobre el usuario.

PRIORIDADES

El kernel utiliza un sistema de asignación de un valor numérico de enteros a los procesos para poder administrar el tiempo de CPU de cada proceso. Cada proceso tiene dos valores de prioridad: estático y dinámico.

- La prioridad estática, mejor conocida como niceness no cambia al menos que el usuario no indique que debe cambiar.
- La prioridad dinámica está basada en la estática, pero el kernel la cambia para distribuir más adecuadamente el tiempo del CPU. Si un proceso está utilizando todo su tiempo asignado de CPU, el kernel modificara la prioridad dinámica para que otros procesos puedan adquirir más tiempo de CPU.

Cuando hablamos de prioridad, casi siempre se habla de prioridad estática ya que no se tiene control sobre la dinámica.

SEÑALES

Las señales son un mecanismo que provee el kernel para que los procesos se intercomunicuen. Existe un número predeterminado de señales (normalmente 32) disponibles.

Se pueden ver también las señales como un tipo de información de centinelas, ya que interrumpen procesos para corregir situaciones anormales.

Las señales tienen significados predefinidos, pero algunas pueden ser programadas, mediante **manejadoras de señales (signal handlers)**.

Dos señales jamás podrán ser ignoradas o programadas: STOP y KILL.

DAEMON o DEMONIO

En sistemas Windows los conocemos como Servicios. Antiguamente en sistemas MS-DOS se les conocía como programas residentes.

Es un tipo especial de proceso no interactivo, es decir, que se ejecuta en segundo plano en vez de ser controlado directamente por el usuario.

BLOQUE II

TEMA 04

2.11 INTERFAZ GRÁFICA DE USUARIO (GUI)

Un aspecto interesante de Linux, a diferencia de Windows y Mac OS X, es su compatibilidad con numerosos entornos de escritorio, lo que ha permitido a los usuarios de escritorio elegir el entorno de escritorio adecuado y más adecuado para trabajar mejor, de acuerdo con sus necesidades informáticas.

Los GUI más populares para Linux son los siguientes:

- GNOME 3
- KDE Plasma 5
- Cinnamon
- MATE
- Xfce
- LXQt
- X Window
- LXDE
- Pantheon Desktop
- Deepin
- Enlightenment
- Budgie
- i3-gaps
- Fluxbox
- Sugar
- Trinity

2.12 SHELL (INTERPRETE DE COMANDOS)

El Shell es el intérprete de comandos en un sistema operativo como Unix o GNU / Linux. Es un programa que ejecuta otros programas. Proporciona al usuario una interfaz para que pueda ejecutar diferentes comandos o utilidades con algunos datos de entrada.

Cuando el Shell ha terminado de ejecutar un programa, envía una salida al usuario en la pantalla, que es el dispositivo de salida estándar. Por esta razón, se le conoce como el " intérprete de comandos".

El Shell es mucho más que un simple intérprete de comandos, también es un lenguaje de programación propio con construcciones de lenguaje de programación completas como ejecución condicional, bucles, variables, funciones y muchos más. Es por eso que el Shell de Unix / GNU Linux es más poderoso en comparación con el Shell de Windows.

Algunos de los Shell más utilizados en Unix / GNU Linux son:

- Bash Shell ≡ Shell de Bourne
- Dash
- Tcsh ≡ C Shell
- Ksh Shell ≡ Korn Shell
- Zsh Shell
- Fish
- Nash
- PowerShell

BLOQUE II

TEMA 04

2.13 COMANDOS

En esta sección se estudian los comandos Linux más usuales, **desde el punto de vista del usuario**, dado que los comandos de administración se verán posteriormente en el Bloque IV.

Dado que la lista es extensa y está sobradamente documentada, se entrega junto con este tema un anexo donde se pueden estudiar los comandos.

Se recomienda estudiar, **como mínimo**, los siguientes comandos:

- NAVEGACION POR EL SISTEMA DE FICHEROS Y DIRECTORIOS \equiv **pwd** , **mkdir**, **rmdir**, **rm**, **cd**, **ls**
- MANEJO DE FICHEROS \equiv **file**, **less**, **cp**, **mv**, **ln**, **head**, **tail**, **cat**, **tac**, **grep**, **touch**, **locate**, **find**, **diff**
- GESTION DE PERMISOS \equiv **chmod**, **chown**, **umask**, **su**, **sudo**, **chgrp**
- COMANDOS DE DISCO \equiv **du**, **df**
- COMANDOS DE DOCUMENTACION \equiv **man**
- COMANDOS DEL EDITOR VI \equiv
- PROCESOS \equiv **ps**, **kill**, **killall**, **Renice**
- SISTEMA \equiv **who**, **w**
- COMPRESION \equiv **tar**, **gzip**, **bzip2** y **zip**

BLOQUE II**TEMA 04****2.14 DISTRIBUCIONES**

Se detalla a continuación una lista de distribuciones de Linux, en general.

Alpine Linux	Knoppix	Slax
ALT Linux	LibreCMC	SliTaz
Arch Linux	Mint	Solus
BLAG	Mageia	SourceMage
Bodhi Linux	Mandriva	SteamOS
Canaima	Manjaro	SUSE Linux
CentOS	MEPIS	Tails
Chakra	Musix	Trisquel
ClearOS	OpenELEC	TurnKey
CoreOS	openSUSE	Ubuntu/Edubuntu
CrunchBang Linux	OpenWrt	Ubuntu GNOME
Damn Small Linux	Parabola	Ubuntu MATE
Debian	Pardus	Kubuntu
Devuan	Parsix	Xubuntu
Dragora	PCLinuxOS	Lubuntu
dyne:bolic	Pentoo	Ututo
Fedora	Porteus	VectorLinux
Funtoo	Puppy Linux	Void Linux
Gentoo	Red Hat	Webconverger
gNewSense	Rxart Desktop	XBMC Live
Guix System	Sabayon Linux	Zentyal
Hyperbola	Scientific Linux	Zenwalk
Kali Linux	Slackware	

Muchas de estas distribuciones Linux están basadas unas en otras. A continuación, se especifica un enlace en la que se puede consultar estas relaciones entre distribuciones.

https://es.wikipedia.org/wiki/Anexo:Distribuciones_Linux

BLOQUE II**TEMA 04****3. SISTEMAS WINDOWS****3.1 INTRODUCCION**

Windows es el nombre de una familia de distribuciones de software para PC, teléfonos inteligentes, servidores y sistemas empujados, desarrollados y vendidos por Microsoft y disponibles para múltiples arquitecturas, tales como x86, x86-64 y ARM.

Windows dispone de distintas versiones, aunque actualmente solo están vigentes (continúan con soporte) las versiones 10 y 11, y para la 10 el soporte es parcial, por lo que en breve se dejará de mantener.

VERSIONES

Windows 11 se puede encontrar en varias versiones distintas, dependiendo del tipo de licencia que se necesite.

Versión	Público objetivo	Características destacadas
Windows 11 Home	Usuarios domésticos	Interfaz moderna, integración con Microsoft Store, Widgets, Snap Layouts, sin funciones empresariales
Windows 11 Pro	Profesionales y pequeñas empresas	Todo lo de Home + BitLocker, Hyper-V, Escritorio remoto, políticas de grupo, cifrado avanzado
Windows 11 Pro for Workstations	Usuarios con hardware avanzado	Soporte para sistemas con múltiples CPUs, ReFS, memoria persistente, rendimiento optimizado
Windows 11 Enterprise	Grandes organizaciones	Todo lo de Pro + seguridad avanzada, control de dispositivos, App Guard, DirectAccess, gestión centralizada
Windows 11 Education	Centros educativos	Similar a Enterprise pero con licencias educativas, herramientas de gestión para aulas y estudiantes
Windows 11 Pro Education	Instituciones académicas	Basado en Pro, con configuraciones específicas para entornos educativos
Windows 11 SE	Dispositivos educativos económicos	Optimizado para estudiantes, apps limitadas, gestión simplificada, pensado para equipos tipo Chromebook

En este tema se estudiará Windows desde el punto de vista de sistema de escritorio de usuario. En el Bloque 4 se estudiará Windows Server y la administración del mismo.

BLOQUE II

TEMA 04

Requisitos del sistema

Resumen:

- Procesador: 1 GHz o más y con 2 o más núcleos. Solo de 64Bits
- Memoria RAM: Mínimo 4GB
- Almacenamiento: Mínimo 64GB
- Firmware: UEFI, compatible con arranque seguro
- TPM: TPM 2.0
- Tarjeta Gráfica: Compatible DirectX 12 o posterior con controlador WDDM 2.0
- Pantalla: Alta definición (720p) y más de 9" en diagonal, con canal de 8 bits por color.

Los requisitos básicos del sistema de Windows 11 son similares a los de las últimas versiones de Windows 10. Sin embargo, Windows 11 **sólo admite sistemas de 64 bits**, como los que utilizan un procesador **x86-64 o ARM64**; Se ha eliminado el soporte para procesadores IA-32.

También se incrementaron los requisitos mínimos de RAM y almacenamiento; Windows 11 ahora requiere **al menos 4 GB de RAM y 64 GB de almacenamiento**. El modo S solo es compatible con la edición Home de Windows 11.

Sólo se admiten procesadores Intel Core de 8.ª Gen, 9.ª Gen, 10.ª Gen y 11.ª Gen y superiores; AMD Zen+ y superiores, y Qualcomm Snapdragon 850 y superiores.

El BIOS heredado ya no es compatible; ahora **se requiere un sistema UEFI** con arranque seguro y un **coprocesador de seguridad TPM 2.0**.

La tecnología de Módulo de plataforma segura (TPM) está diseñada para proporcionar funciones basadas en hardware y relacionadas con la seguridad. Un chip TPM es un procesador criptográfico seguro diseñado para llevar a cabo operaciones criptográficas. El chip incluye varios mecanismos de seguridad físicos para que sea resistente a alteraciones y el software malintencionado no pueda alterar las funciones de seguridad del TPM.

Algunas de las principales ventajas de usar la tecnología TPM son:

- Generar, almacenar y limitar el uso de claves criptográficas.
- Usar la tecnología TPM para la autenticación de dispositivos de plataforma mediante clave RSA única de TPM, que se graba dentro.
- Garantizar la integridad de la plataforma llevando y almacenando medidas de seguridad.

Información técnica

- Núcleo: NT 10.0
- Tipo de núcleo: Híbrido
- Plataformas admitidas: x86-64 y ARM
- Interfaz gráfica: Microsoft Fluent Design
- Método de actualización: Windows Update

BLOQUE II

TEMA 04

Interface gráfica

Microsoft Fluent Design, conocido antes como Project Neon, es la interfaz gráfica incluida en el sistema operativo **Windows 10** y **Windows 11**. La interfaz pretende mejorar el aspecto de Windows 10 proporcionando un diseño intuitivo y fluido con características de transparencia similares a Windows Aero (interface gráfica de Windows 7 y Windows Vista), pero dándole mayor énfasis a las interfaces táctiles.

Principales novedades de Windows 11 frente a Windows 10

Interfaz y experiencia de usuario

- Barra de tareas centrada y rediseñada.
- Menú Inicio simplificado, sin Live Tiles (iconos animados que se actualizan en tiempo real).
- Esquinas redondeadas y efectos de transparencia (efecto Mica).
- Snap Layouts y Snap Groups para organizar ventanas con más precisión.
- Widgets rediseñados con contenido personalizado y acceso desde pantalla de bloque

Integración de inteligencia artificial

- **Copilot** en Windows: asistente basado en IA integrado en el sistema, accesible desde la barra de tareas.
- Acciones rápidas de imagen en el Explorador de archivos: desenfocar, eliminar fondo, buscar con Bing Visual Search.
- Resumen de archivos en OneDrive y SharePoint usando IA.

Seguridad y rendimiento

- Requisitos de hardware más estrictos: TPM 2.0, Secure Boot, CPU moderna.
- Mejor gestión de actualizaciones: más rápidas y menos intrusivas.
- Protección de identidad y cifrado mejorado para cuentas Microsoft y empresariales.

Gaming y multimedia

- Auto HDR y DirectStorage para mejorar tiempos de carga y gráficos en juegos.
- Mejor soporte para monitores de alta tasa de refresco y configuraciones multimonitor.

Explorador de archivos y productividad

- Explorador de archivos con pestañas.
- Integración con OneDrive más profunda.
- Mejoras en el portapapeles, historial y sincronización entre dispositivos.

BLOQUE II

TEMA 04

Cambios en cuentas y configuración

- Inicio de sesión con cuenta Microsoft obligatorio en Home.
- Configuración más intuitiva, con paneles reorganizados.
- Visor de Braille y mejoras en accesibilidad para Narrador.

Funciones eliminadas o modificadas

- Cortana ya no está integrada como asistente principal.
- Internet Explorer completamente retirado.
- Barra de tareas menos personalizable (no se puede mover ni agrupar como antes).
- Modo tablet clásico eliminado

3.1 GESTORES DE ARRANQUE

Un bootloader o gestor de arranque es un software especial que carga en la memoria interna el sistema operativo instalado en el sistema.

Como se puede apreciar en esa tabla, los gestores de arranque principales en entorno Windows son:

BOOTMGR (Windows Boot Manager) ≡ Windows Vista en adelante. Carga el sistema operativo desde el archivo BCD. Compatible con UEFI y BIOS.

BCD (Boot Configuration Data) ≡ Windows Vista en adelante. Base de datos que reemplaza el antiguo boot.ini. Se gestiona con la aplicación BCDEDIT.

NTLDR ≡ Windows XP y anteriores (Antiguo). Cargador de arranque clásico. Usa boot.ini para configurar el arranque.

GRUB Sistemas duales (Linux + Windows) ≡ Detecta múltiples sistemas operativos. Se instala desde Linux y puede iniciar Windows.

EasyBCD ≡ Windows Vista en adelante. Interfaz gráfica para modificar BCD. Ideal para usuarios que no quieren usar comandos. Permite arranque dual (más de un S.O instalado en el disco)

UEFI Boot Manager ≡ Windows 8 en adelante (modo UEFI). Usa entradas en la NVRAM para gestionar el arranque. Compatible con Secure Boot.

BLOQUE II

TEMA 04

Bootmgr

NTLDR fue sustituido por Bootmgr a partir de la versión 6 de la familia de sistemas Windows NT (Windows Vista y posteriores).

El sector de arranque o UEFI carga el Administrador de arranque de Windows (un archivo llamado BOOTMGR en el sistema o en la partición de arranque), accede al almacén de datos de configuración de arranque (BCD) y usa la información para cargar el sistema operativo. Luego, el BCD invoca el cargador de arranque y, a su vez, procede a iniciar el kernel de Windows. La inicialización en este punto procede de manera similar a las versiones anteriores de Windows NT.

Boot Configuration Data (BCD) es una base de datos independiente del firmware para la configuración del proceso de arranque en sistemas operativos Windows de Microsoft, reemplaza al fichero boot.ini usado en NTLDR.

El BCD es almacenado en un archivo de datos que tiene el mismo formato que el registro de Windows. Además, son cifrados de forma que no puedan ser editados manualmente o mediante un editor de texto.

- Para el arranque en particiones UEFI, el archivo se encuentra en /EFI/Microsoft/Boot/BCD en la partición del sistema EFI.
- Para el arranque de BIOS tradicional, el archivo está en /boot/BCD en la partición activa.

La configuración del BCD puede ser modificada usando:

- Bcdedit (desde la línea de comandos)
- Usando el Registro de Windows (regedit.exe)
- Usando WMI (Windows Management Instrumentation)
- Con aplicaciones de terceros como: EasyBCD, BOOTICE, Visual BCD Editor, Clover EFI bootloader (permite instalar Linux o Windows en sistemas Mac), rEFInd Boot Manager (para sistemas EFI y UEFI), BootIt (compatible con todos los sistemas operativos).

6.0	Windows Vista
	Windows Server 2008
6.1	Windows 7
	Windows Server 2008 R2
	Windows Home Server 2011
	Windows Thin PC
6.2	Windows 8
	Windows Server 2012
6.3	Windows 8.1
	Windows Server 2012 R2
10.0	Windows 10
	Windows Server 2016
	Windows Server 2019

BLOQUE II

TEMA 04

3.1 PARTICIONES EN WINDOWS

En el entorno Windows, existe una utilidad llamada **DISKPART**, usada como línea de comandos, que permite la gestión de discos, particiones, volúmenes, discos virtuales.

DiskPart reemplazó a **FDISK** como herramienta de particionado en Windows a partir de Windows 2000. Mientras FDISK era limitado y solo funcionaba en sistemas FAT/MBR, DiskPart ofrece soporte para discos dinámicos, particiones GPT, scripting y entornos UEFI.

¿Qué permite hacer DiskPart?

- Crear, eliminar y formatear particiones.
- Convertir discos entre MBR y GPT.
- Asignar letras de unidad.
- Limpiar discos completamente (clean y clean all).
- Administrar discos virtuales (VHD/VHDX).
- Automatizar tareas con scripts

Antes de poder usar los comandos de diskpart, se debe seleccionar un objeto para darle foco. Una vez que un objeto tiene el foco, cualquier comando de diskpart que escriba actuará sobre ese objeto.

Lista de objetos disponibles

Se pueden enumerar los objetos disponibles y determinar el número o la letra de unidad de un objeto mediante:

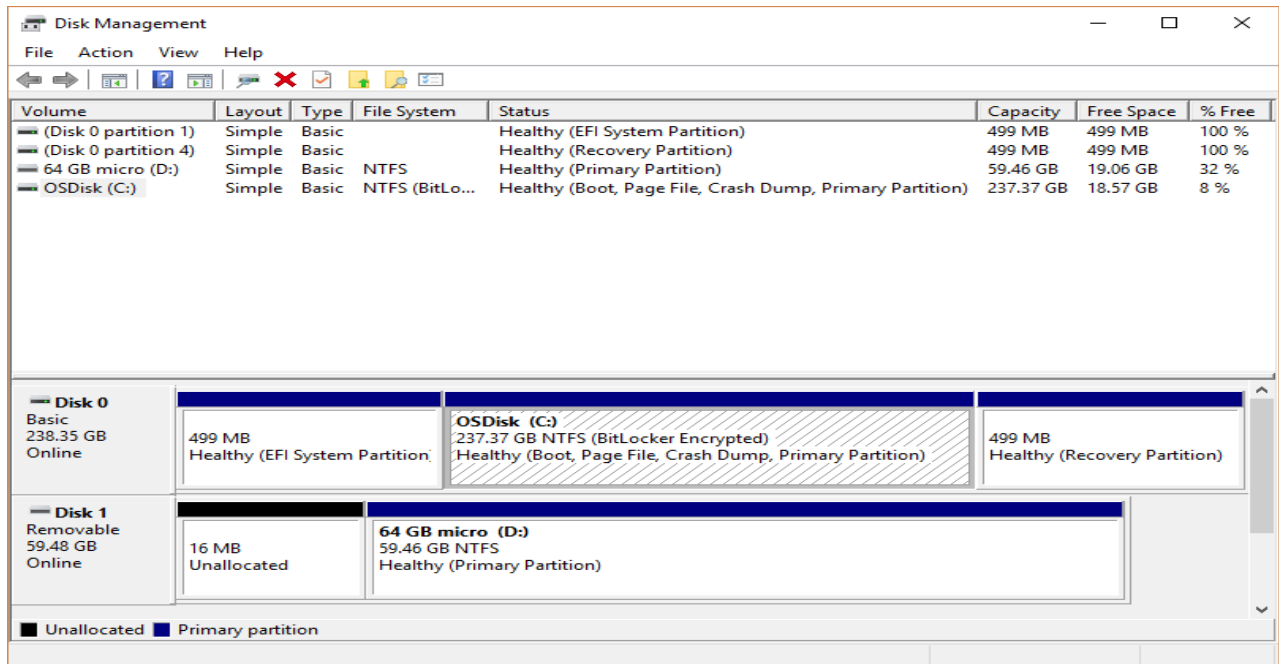
- list disk - Muestra todos los discos.
- list volume - Muestra todos los volúmenes.
- list partition - Muestra las particiones en el disco que tienen el foco.
- list vdisk - Muestra todos los discos.

Para ejecutar diskpart, se deben tener permisos de administrador o pertenecer a un grupo de usuarios con permisos de administrador.

Existe otra utilidad, que se usa mediante interface gráfico, llamada **Administrador de discos** (o Disk Management) que permite igualmente la gestión de particiones y volúmenes.

BLOQUE II

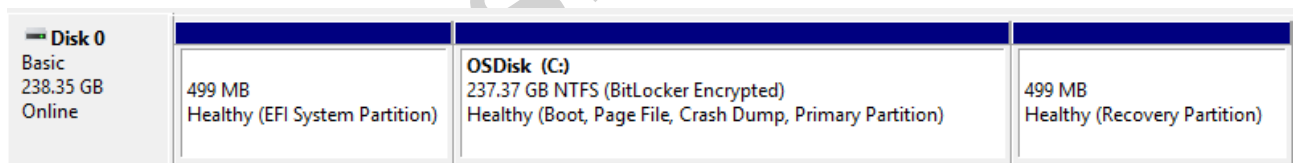
TEMA 04



Entre otras cosas, esta utilidad permite:

- Inicializar un nuevo disco (Formatear)
- Extender un volumen usando espacio libre que exista en el mismo disco
- Reducir el tamaño de una partición.
- Cambiar letra asignada a una unidad o asignar una nueva letra de unidad

Windows normalmente incluye tres particiones en su unidad principal.



- Partición del sistema EFI: En ordenadores modernos, utilizada para arrancar el ordenador y el sistema operativo.
- Unidad del sistema operativo Windows (C:\): Aquí es donde se instala Windows y, por lo general, donde coloca el resto de sus aplicaciones y archivos.
- Partición de recuperación: Se almacenan las herramientas especiales para ayudar a recuperar Windows en caso de que tenga problemas para iniciar o se encuentre con otros problemas graves.

BLOQUE II

TEMA 04

3.2 MEMORIA VIRTUAL

La paginación es una parte importante de las implementaciones de memoria virtual en los sistemas operativos modernos, ya que utiliza almacenamiento secundario para permitir que los programas excedan el tamaño de la memoria física disponible, que recupera datos del almacenamiento secundario para usar en la memoria principal. El sistema operativo recupera datos del almacenamiento secundario en bloques del mismo tamaño llamados páginas.

El archivo utilizado para la paginación en la familia de Windows es **pagefile.sys**. La ubicación predeterminada del archivo de paginación está en el directorio raíz de la partición donde está instalado Windows y tiene el estado de **oculto y protegido por el sistema**.

Windows puede configurarse para utilizar espacio libre en **cualquier unidad** disponible para archivos de paginación. Sin embargo, es necesario que la partición de arranque (es decir, la unidad que contiene el directorio de Windows) tenga un archivo de paginación si el sistema está configurado para escribir volcados de memoria completa o del kernel después de un pantallazo azul (o pantalla de la muerte).

Windows usa el archivo de paginación como almacenamiento temporal para el volcado de memoria. Cuando se reinicia el sistema, Windows copia el volcado de memoria del archivo de paginación en un archivo separado y libera el espacio que se utilizó en el archivo de paginación.

Características

- El archivo pagefile.sys se puede eliminar
- Windows permite desactivar el uso de este archivo.
- Permite cambiar el tamaño asignado al archivo pagefile.sys

3.3 ARCHIVOS Y DIRECTORIOS IMPORTANTES

A continuación, se detalla los directorios más importantes que se pueden encontrar en una instalación de Windows:

C:\Windows ≡ Contiene el núcleo del sistema operativo, DLLs, drivers, y servicios.

C:\Program Files o C:\Archivos de Programas ≡ Instalación de aplicaciones de 64 bits.

C:\Program Files (x86) o C:\Archivos de Programas (x86) ≡ Instalación de aplicaciones de 32 bits.

C:\Users o C:\Usuarios ≡ Perfiles de usuario: documentos, escritorio, configuración personal.

C:\ProgramData ≡ Datos compartidos entre usuarios y aplicaciones. Oculto por defecto.

C:\Recycle.Bin ≡ Contenedor de archivos eliminados (papelera de reciclaje).

C:\System Volume Information ≡ Restauración del sistema, puntos de recuperación. Oculto y protegido. Se encuentra tanto en discos internos como externos.

BLOQUE II

TEMA 04

En cuanto a los archivos de sistema más significativos, podemos encontrar:

pagefile.sys ≡ Ubicado en la raíz (C:\). Archivo de paginación (memoria virtual).

hiberfil.sys ≡ Archivo de hibernación. Oculto. Ubicado en la raíz (C:\). Guardar el estado de la sesión (programas abiertos, documentos, procesos en ejecución) cuando el equipo entra en hibernación. Se crea automáticamente cuando el modo de hibernación está activado. Su tamaño suele ser cercano al total de la memoria RAM instalada. Se puede eliminar si se desactiva la hibernación.

A diferencia del **modo suspensión**, la hibernación no consume energía, ya que el equipo se apaga completamente.

desktop.ini ≡ Ubicado en carpetas de usuario. Configura la apariencia de carpetas en el Explorador.

Thumbs.db ≡ Ubicado en carpetas con imágenes. Base de datos de miniaturas. Puede eliminarse sin afectar el sistema.

3.4 CUENTAS DE USUARIO

Windows 11 ofrece dos tipos principales de cuentas de usuario: **cuentas locales y cuentas de Microsoft**. Cada una puede tener permisos de administrador o estándar, lo que define su nivel de control sobre el sistema.

Tipos de permisos:

Administrador ≡ Control total del sistema: instalar software, cambiar configuración global, crear usuarios.

Estándar ≡ Uso limitado: ejecutar aplicaciones, cambiar configuración personal, pero no instalar ni modificar el sistema.

Cuenta local

Una cuenta local es un perfil de usuario que existe únicamente en el equipo donde se crea. No está vinculada a ningún servicio en línea (Por ejemplo: Microsoft Account) y no requiere conexión a Internet para funcionar.

- Usuario + contraseña definidos localmente.
- No requiere conexión a Internet ni correo electrónico.
- Guarda configuración y archivos solo en el dispositivo.
- Ideal para uso privado, equipos compartidos o entornos sin conexión.
- Puede ser administrador o usuario estándar.

BLOQUE II

TEMA 04

Cuenta de Microsoft

Una cuenta de Microsoft es una credencial única que permite acceder a múltiples servicios de Microsoft, como Windows, Office, OneDrive, Xbox, Teams y más. Está vinculada a un correo electrónico (Outlook, Hotmail, Gmail, etc.) y permite sincronizar datos entre dispositivos.

- Vinculada a un correo electrónico (Outlook, Hotmail, etc.).
- Permite sincronizar configuración, historial, contraseñas y archivos entre dispositivos.
- Requiere conexión a Internet para funciones avanzadas.
- Necesaria para usar Microsoft Store, OneDrive, y otras apps integradas.
- También puede ser administrador o usuario estándar.

Otros tipos en de cuentas

Cuenta profesional o educativa ≡ asociada a dominios empresariales o escolares (Azure AD o Microsoft Entra ID).

Cuenta infantil ≡ vinculada a una cuenta familiar, con controles parentales.

Cuenta invitada (obsoleta) ≡ existía en versiones anteriores, pero ya no está disponible por defecto en Windows 11.

3.4.2 CONTROL DE CUENTAS DE USUARIO (UAC)

El Control de Cuentas de Usuario (UAC - User Account Control) en Windows 11 es una función de seguridad que protege el sistema contra cambios no autorizados, solicitando confirmación antes de ejecutar acciones que requieren privilegios de administrador.

- UAC es una **capa de seguridad** que evita que aplicaciones o usuarios realicen cambios en el sistema sin autorización.
- Se activa cuando una acción requiere privilegios elevados, como instalar software, modificar configuraciones del sistema o acceder a archivos protegidos.
- Previene la ejecución silenciosa de malware al requerir intervención del usuario.

Cuando una aplicación intenta realizar un cambio sensible, UAC muestra una ventana emergente que pregunta si se desea permitir la acción:

- Si se está usando una cuenta estándar, se pedirá ingresar credenciales de administrador.
- Si se está usando una cuenta de administrador, solo se debe confirmar la acción

BLOQUE II

TEMA 04

El sistema UAC, es importante, desde el punto de vista de la seguridad, por motivos como los siguientes:

- Protección contra malware y ransomware: impide que software malicioso se ejecute con privilegios elevados sin tu consentimiento.
- Control administrativo: útil en entornos corporativos o educativos para evitar que los usuarios alteren configuraciones críticas.
- Auditoría y trazabilidad: en sistemas multiusuario, permite saber cuándo y quién autorizó cambios.

Se puedes ajustar, activar o desactivar el comportamiento del UAC desde:

- Panel de control → Sistema y seguridad → Cambiar configuración del Control de cuentas de usuario.

Así mismo, para activar o desactivar el sistema UAC, existen varios métodos:

- Desde el panel de control (mencionado anteriormente).
- Usando el editor del registro (regedit).
- Mediante directivas de grupo.
- Usando PowerShell o CMD.

3.4.3 AUTENTICACION DE USUARIO. WINDOWS HELLO

Windows Hello es una función de seguridad que viene incluida en todas las ediciones de Windows desde su lanzamiento. Lo que hace esta función de seguridad es permitirnos olvidarnos de la contraseña y utilizar otros métodos de autenticación más rápidos y cómodos para iniciar sesión en el ordenador.

Dicho de otro modo, Windows Hello es un sistema de **autenticación biométrica** que permite iniciar sesión de forma rápida y segura usando tu rostro, huella dactilar o un PIN local, sin necesidad de contraseñas tradicionales.

Importante: El PIN de Windows Hello no es como una contraseña tradicional. Está vinculado al dispositivo y no se transmite por red, lo que lo hace más seguro frente a ataques remotos.

Privacidad y seguridad

- Los datos biométricos se almacenan localmente en el dispositivo, en un entorno seguro (Trusted Platform Module).
- No se sincronizan ni se envían a Microsoft ni a la nube.
- Puedes mejorar la seguridad activando bloqueo dinámico, que bloquea el equipo si te alejas con un dispositivo Bluetooth emparejado.

Además de Windows Hello, existen otras formas de iniciar sesión en Windows:

- Contraseña: permite usar la clave de la cuenta Microsoft para iniciar sesión.
- Contraseña de imagen: configura una imagen y, al hacer clic en un punto concreto de la misma, se iniciará sesión.
- Clave de seguridad FIDO2: Dispositivo físico USB/NFC para autenticación.

BLOQUE II

TEMA 04

- Bloqueo dinámico. Bloquea el equipo si te alejas con un dispositivo Bluetooth emparejado.

3.5 REGISTRO DE WINDOWS

El registro de Windows es una **base de datos jerárquica centralizada** que se usa en sistemas Windows para almacenar información necesaria para:

- Configuración del sistema operativo
- Preferencias de usuario
- Parámetros de hardware
- Opciones de software instalado

Se accede mediante el comando *regedit*, que abre el Editor del Registro. También pueden usarse archivos REG, (también conocidos como entradas de registro), que son archivos legibles de texto para importar y exportar partes del registro.

¿Para qué sirve?

- Personalización avanzada del sistema (por ejemplo, ocultar elementos del escritorio, modificar el comportamiento de menús).
- Solución de problemas: restaurar funciones dañadas, eliminar rastros de malware, corregir errores de configuración.
- Automatización y administración: en entornos corporativos, permite aplicar políticas mediante scripts o directivas de grupo.
- Control de arranque y servicios: gestiona qué procesos se inician con Windows.

El registro contiene dos elementos básicos: **claves y valores**.

Las claves del registro son similares a carpetas: además de los valores, cada clave puede contener subclaves, que a su vez pueden contener más subclaves, y así sucesivamente. Las claves están referenciadas con una sintaxis parecida a los nombres de las rutas de Windows, y usan barras diagonales inversas para indicar los distintos niveles jerárquicos. Cada subclave tiene obligatoriamente un nombre: una cadena que no puede contener barras diagonales inversas y en la que no se distingue entre mayúsculas y minúsculas. El tamaño máximo de un nombre de clave es de 255 caracteres.

Existen una serie de claves ya predefinidas:

Clave raíz	Contenido
HKEY_CLASSES_ROOT (HKCR)	Asociaciones de archivos y tipos de objetos COM.
HKEY_CURRENT_USER (HKCU)	Configuración del usuario actual.
HKEY_LOCAL_MACHINE (HKLM)	Configuración global del sistema y hardware.
HKEY_USERS (HKU)	Perfiles de todos los usuarios.
HKEY_CURRENT_CONFIG (HKCC)	Información de configuración activa del hardware.

BLOQUE II**TEMA 04**

Los valores del registro son pares de nombres y datos almacenados dentro de las claves. Cada valor de registro almacenado en una clave de registro posee un nombre único, sin distingue entre mayúsculas y minúsculas.

Cada valor puede almacenar datos arbitrarios de longitud y codificado variables, pero asociados a un tipo simbólico de datos (definido como una constante numérica) que expresa cómo analizar estos datos. Los tipos estándar son:

Lista de tipos de valores estándar del registro		
Nombre	Nombre de tipo simbólico de datos	Significado y codificación de los datos almacenados en el valor de registro
0	REG_NONE	Datos sin ningún tipo (en todo caso, el valor almacenado)
1	REG_SZ	Valor de cadena, normalmente almacenado y mostrado en UTF-16LE (cuando se utiliza la versión Unicode de las funciones API de Win32), que generalmente termina con un carácter nulo
2	REG_EXPAND_SZ	Valor de cadena "expandible" que puede contener variables de entorno, normalmente almacenado y mostrado en UTF-16LE, que generalmente termina con un carácter nulo
3	REG_BINARY	Datos binarios (cualquier dato arbitrario)
4	REG_DWORD / REG_DWORD_LITTLE_ENDIAN	Valor DWORD, número entero no negativo de 32 bits (números entre el 0 y el 4.294.967.295 [232 – 1]) (little-endian)
5	REG_DWORD_BIG_ENDIAN	Valor DWORD, número entero no negativo de 32 bits (números entre el 0 y el 4.294.967.295 [232 – 1]) (big-endian)
6	REG_LINK	Enlace simbólico (UNICODE) a otra clave de registro, especificando una clave raíz y la ruta a la clave objetivo
7	REG_MULTI_SZ	Valor de cadena múltiple, que generalmente es una lista ordenada de cadenas no vacías, normalmente almacenadas y mostradas en UTF-16LE, cada una de ellas terminada en un carácter nulo, y la lista normalmente también termina con un carácter nulo.
8	REG_RESOURCE_LIST	Lista de recursos (usada por la enumeración y configuración del hardware Plug-n-Play)
9	REG_FULL_RESOURCE_DESCRIPTOR	Descriptor de recursos (usado por la enumeración y configuración del hardware Plug-n-Play)
10	REG_RESOURCE_REQUIREMENTS_LIST	Lista de requisitos de recursos (usada por la enumeración y configuración del hardware Plug-n-Play)
11	REG_QWORD / REG_QWORD_LITTLE_ENDIAN	Valor QWORD, número entero de 64 bits (puede ser big-endian o little-endian, o sin especificar). (Introducido en Windows XP)

3.6 SEGURIDAD EN WINDOWS

Windows 11 incorpora múltiples capas de seguridad que van desde protecciones del sistema operativo hasta herramientas para el usuario final, incluyendo cifrado, aislamiento, control de acceso, y defensa contra amenazas.

Aquí tienes un resumen estructurado de los principales componentes y conceptos relacionados con la seguridad en Windows 11:

Control de cuentas de usuario (UAC) ≡ Explicado anteriormente.

SmartScreen ≡ Bloquea sitios web maliciosos y descargas sospechosas. Smart Screen es una capa intermedia entre el usuario y el sistema operativo que nos permite controlar todo lo que se ejecuta en nuestro PC. Esta capa funciona sobre todo cuando intentamos abrir un programa o archivo que ha sido descargado de Internet o que ha llegado a través de cualquier otro programa.

También es capaz de protegernos mientras estamos navegando por Internet desde el explorador Microsoft Edge. SmartScreen cuenta con una base de datos dinámica gracias a la cual puede identificar las webs que puedan ser peligrosas, sospechosas o utilizadas para llevar a cabo ataques de phishing y, en caso de intentar visitar alguna, la bloqueará.

BLOQUE II

TEMA 04

Windows Defender Antivirus ≡ (ahora llamado Microsoft Defender Antivirus) es la solución de seguridad integrada en Windows 11 que protege, en tiempo real, contra virus, malware, ransomware y otras amenazas sin necesidad de instalar software adicional.

- Utiliza inteligencia artificial, análisis de macrodatos y la nube de Microsoft para detectar amenazas emergentes y sofisticadas.
- Se actualiza automáticamente a través de Windows Update, manteniendo la base de datos de virus al día.
- Integra protección contra ransomware, control de acceso a carpetas y análisis de comportamiento para bloquear ataques antes de que ocurran.
- Compatible con otras soluciones antivirus, aunque se desactiva automáticamente si se instala otro antivirus principal.
- Incluye protección web mediante SmartScreen, que bloquea sitios maliciosos y descargas sospechosas.

Firewall de Windows ≡ Junto a Windows Defender, Windows también nos ofrece un completo y robusto firewall. Por defecto viene configurado para que, junto a las demás medidas de seguridad, pueda protegernos de prácticamente cualquier amenaza sin tener que preocuparnos por su configuración.

El Firewall de Windows 11 (Microsoft Defender Firewall), es una barrera esencial que protege tu equipo controlando el tráfico de red entrante y saliente. Su correcta configuración es clave para prevenir accesos no autorizados y ataques remotos.

- Filtra el tráfico de red según reglas predefinidas o personalizadas.
- Bloquea conexiones no autorizadas y permite solo aquellas que cumplen criterios seguros.
- Funciona en tres perfiles de red: Privada, Pública y de Dominio, cada uno con reglas distintas.
- Integración con AV Defender. Trabaja junto al antivirus para bloquear amenazas en red.
- Consola avanzada. A través de *wf.msc* puedes gestionar reglas detalladas, exportarlas o importarlas.

Bitlocker ≡ Característica de protección de la información que ayuda a mitigar el acceso no autorizado a datos mejorando las protecciones de archivo y de sistema. Nos permite **cifrar** fácilmente cualquier disco duro, o unidad externa, que conectemos al ordenador. De esta manera, si alguien intenta acceder a nuestros datos y no tiene la correspondiente contraseña de Windows, no podrá acceder a ellos.

Usa algoritmos de cifrado avanzado (como AES-XTS de 128 o 256 bits).

Requisitos técnicos

- Edición compatible: Solo disponible en Windows 11 Pro, Enterprise y Education.
- TPM 2.0 (Trusted Platform Module): para almacenar claves de forma segura.
- UEFI y Secure Boot: recomendados para una protección completa.
- En equipos sin TPM, se puede usar BitLocker con una unidad flash USB como clave de inicio, aunque con menor seguridad.

No está disponible en Windows 11 Home, aunque algunos equipos compatibles pueden usar Cifrado de dispositivo, una versión simplificada.

Funciones clave

BLOQUE II

TEMA 04

- Cifrado de disco completo \equiv Protege todo el contenido de la unidad, incluyendo archivos del sistema.
- Cifrado de unidades externas (BitLocker To Go) \equiv Permite cifrar memorias USB y discos externos.
- Autenticación previa al arranque \equiv Solicita PIN o clave antes de iniciar Windows.
- Recuperación con clave o archivo \equiv Si se pierde el acceso, se puede recuperar con una clave de 48 dígitos o archivo de recuperación.

Si el dispositivo pide la clave de recuperación de BitLocker, la siguiente información, es necesario tenerla almacenada en algún sitio. Estos son algunos lugares a comprobar para encontrar la clave:

- Cuenta Microsoft \equiv Se guarda automáticamente si activas BitLocker.
- Archivo en disco o USB \equiv Puedes guardar la clave como archivo .txt en otra unidad o pendrive. No debe estar en la misma unidad cifrada.
- Impresión física \equiv Puedes imprimir la clave y guardarla en un lugar seguro (caja fuerte, archivador).
- Active Directory / Azure AD \equiv En empresas, las claves se almacenan automáticamente en el directorio para recuperación centralizada.
- Administrador de contraseñas \equiv Puedes copiar la clave en un gestor seguro como Bitwarden, KeePass, etc.

Crear Puntos de Restauración \equiv Piensa en esta función como un punto de guardado para tu dispositivo. Si algo sale mal mientras aumentas la seguridad de tu equipo, siempre puedes volver al punto de instalación anterior y empezar desde cero. Dado que esta función está desactivada de manera predeterminada en Windows 10, se debe activar manualmente.

3.7 POWERSHELL

PowerShell (originalmente llamada Windows PowerShell) es una interfaz de consola (CLI) con posibilidad de escritura y unión de comandos por medio de instrucciones (scripts). Originalmente, esta herramienta fue denominada como MONAD.

Requiere de la instalación previa del framework .NET versión 2.0 para su funcionamiento. Se presentó junto con el sistema operativo Windows Vista y se incluye también en Windows 7, Windows 8 y Windows 10. La última versión de PowerShell es la **Versión 7**

También puede ser instalado en sistemas Linux y MacOS.

Powershell no solo permite interactuar con el sistema operativo, sino también con programas de Microsoft como SQL Server, Exchange o IIS. La principal utilidad de Powershell es permitir automatizar tareas administrativas al usuario.

La característica distintiva de PowerShell es que es un intérprete de comandos **orientado a objetos**. La información de entrada y de salida en cada etapa del proceso (**cmdlet**,

BLOQUE II

TEMA 04

"**comándulo**") es un conjunto de instancias de objeto, a diferencia de lo que ocurre con los intérpretes de comandos tradicionales, que solo devuelven y reciben texto.

Un **cmdlet** es un comando ligero que se usa en el entorno de PowerShell. El tiempo de ejecución de PowerShell invoca estos cmdlets en el contexto de los scripts de automatización que se proporcionan en la línea de comandos. El entorno de tiempo de ejecución de PowerShell también los invoca mediante programación a través de las API de PowerShell.

Diferencias entre los cmdlets y los comandos

- Los cmdlets se diferencian de los comandos en otros entornos de Shell de comandos de las siguientes maneras:
- Los cmdlets son instancias de clases .NET; no son ejecutables independientes.
- Los cmdlets se pueden crear desde tan solo una docena de líneas de código.
- Los cmdlets no realizan normalmente su propio análisis, presentación de errores o formato de salida. El tiempo de ejecución de PowerShell administra el análisis, la presentación de errores y el formato de salida.
- Los cmdlets procesan los objetos de entrada de la canalización en lugar de los flujos de texto, y los cmdlets suelen ofrecer objetos como salida a la canalización.
- Los cmdlets están orientados a registros porque procesan un solo objeto cada vez.

3.8 ATAJO DE TECLADO

Usando la tecla Windows ()

El uso de la Tecla Windows para muchos usuarios se limita a pulsar y ver cómo se abre o cierra el menú Inicio, pero sus funciones se amplían si la combinamos con otras teclas.

- Tecla Windows: Abrir o cerrar Inicio.
- Tecla Windows + A: Abrir el Centro de actividades.
- Tecla Windows + B: Llevar el foco al área de notificación.
- Tecla Windows + C: Abrir Cortana en modo de escucha. Este acceso está desactivado de forma predeterminada.
- Tecla Windows + Mayús + C: Abrir el menú de botones de acceso.
- Tecla Windows + D: Mostrar y ocultar el escritorio.
- Tecla Windows + Alt + D: Mostrar y ocultar la fecha y hora en el escritorio.
- Tecla Windows + E: Abrir el Explorador de archivos.
- Tecla Windows + F: Abrir el Centro de opiniones y hacer una captura de pantalla.
- Tecla Windows + G: Abrir la barra de juegos con un juego abierto.
- Tecla Windows + H: Iniciar dictado.
- Tecla Windows + I: Abrir Configuración.
- Tecla Windows + J: Establecer el foco en una sugerencia de Windows cuando haya una disponible.
- Tecla Windows + K: Abrir la acción rápida Conectar.
- Tecla Windows + L: Bloquear el equipo o cambiar de cuenta.
- Tecla Windows + M: Minimizar todas las ventanas.

BLOQUE II

TEMA 04

- Tecla Windows + O: Bloquear la orientación del dispositivo.
- Tecla Windows + P: Elegir un modo de presentación.
- Tecla Windows + Ctrl + Q: Abrir Asistencia rápida.
- Tecla Windows + R: Abrir el cuadro de diálogo Ejecutar
- Tecla Windows + S: Abrir la búsqueda.
- Tecla Windows + Mayús + S: Realizar una captura de pantalla de parte de la pantalla.
- Tecla Windows + T: Desplazarse por las aplicaciones de la barra de tareas.
- Tecla Windows + U: Abrir el Centro de accesibilidad.
- Tecla Windows + V: Abrir el Portapapeles. Para activar este acceso directo, selecciona Inicio > Configuración > Sistema > Portapapeles y activa el botón de alternancia situado debajo de Historial del portapapeles.
- Tecla Windows + Mayús + V: Desplazarse por las notificaciones.
- Tecla Windows + X: Abrir el menú Vínculo rápido.
- Tecla Windows + Y: Cambiar entrada entre Windows Mixed Reality y el escritorio.
- Tecla Windows + Z: Mostrar los comandos disponibles en una aplicación en modo de pantalla completa.
- Tecla Windows + punto (.) o punto y coma (;): Abrir el panel de emojis.
- Tecla Windows + coma (,): Ojear temporalmente el escritorio.
- Tecla Windows + Pausa: Mostrar el cuadro de diálogo Propiedades del sistema
- Tecla Windows + Ctrl + F: Buscar equipos (en una red).
- Tecla Windows + Mayús + M: Restaurar las ventanas minimizadas en el escritorio.
- Tecla Windows + número: Abrir el escritorio e iniciar la aplicación anclada a la barra de tareas en la posición indicada por el número. Si la aplicación ya se está ejecutando, cambiar a esa aplicación.
- Tecla Windows + Mayús + número: Abrir el escritorio e iniciar una nueva instancia de la aplicación anclada a la barra de tareas en la posición indicada por el número.
- Tecla Windows + Ctrl + número: Abrir el escritorio y cambiar a la última ventana activa de la aplicación anclada a la barra de tareas en la posición indicada por el número.
- Tecla Windows + Alt + número: Abrir el escritorio y abrir la lista de accesos directos de la aplicación anclada a la barra de tareas en la posición indicada por el número.
- Tecla Windows + Ctrl + Mayús + número: Abrir el escritorio y abrir una nueva instancia de la aplicación ubicada en la posición determinada en la barra de tareas como administrador.
- Tecla Windows + Tabulador: Abrir la Vista de tareas.
- Tecla Windows + Flecha arriba: Maximizar la ventana.
- Tecla Windows + Flecha abajo: Quitar la aplicación actual de la pantalla o minimizar la ventana del escritorio.
- Tecla Windows + Flecha izquierda: Maximizar la ventana de la aplicación o del escritorio en el lado izquierdo de la pantalla.
- Tecla Windows + Flecha derecha: Maximizar la ventana de la aplicación o del escritorio en el lado derecho de la pantalla.
- Tecla Windows + Inicio: Minimizar todo excepto la ventana del escritorio activa (repetir la acción para restaurar todas las ventanas).
- Tecla Windows + Mayús + Flecha arriba: Expandir la ventana del escritorio a la parte superior e inferior de la pantalla.
- Tecla Windows + Mayús + Flecha abajo: Restaurar/minimizar las ventanas del escritorio activas verticalmente conservando el ancho.

BLOQUE II

TEMA 04

- Tecla Windows + Mayús + Flecha izquierda o derecha: Mover una aplicación o ventana del escritorio de un monitor a otro.
- Tecla Windows + Barra espaciadora: Cambiar entre el idioma de entrada y la distribución del teclado.
- Tecla Windows + Ctrl + Barra espaciadora: Cambiar a una entrada seleccionada anteriormente.
- Tecla Windows + Ctrl + Entrar: Activar Narrador.
- Tecla Windows + Más (+): Abrir la Lupa.
- Tecla Windows + barra diagonal (/): Iniciar la reconversión de IME.
- Tecla Windows + Ctrl + V: Abrir notificaciones animadas.
- Tecla Windows + PrtScr: Toma una captura de pantalla y la envía directo a la carpeta de imágenes.
- Tecla Windows + G: Inicia la aplicación DVR para grabar la pantalla.
- Tecla Windows + Alt + G: Inicia la grabación de pantalla en la ventana que te encuentras.
- Tecla Windows + Alt + R: Cambia entre modo de pantalla secundaria si tienes otro monitor conectado.

Otros atajos que también son muy útiles para copiar y pegar, seleccionar más rápidamente frases o párrafos enteros, etc. Son los siguientes:

- Tecla Ctrl + V o Shift + Insert: Pega un texto en la posición que esté el cursor.
- Tecla Ctrl + C o Ctrl + Insert: Copia en el portapapeles el texto seleccionado.
- Tecla Ctrl + X: Corta en el portapapeles el texto seleccionado.
- Tecla Ctrl + A: Selecciona todo el texto de la página.
- Tecla Ctrl + F: Abre un cuadro para buscar en la página el texto que escribas en él.
- Shift + flechas de dirección: Mueve el cursor por el texto seleccionándolo. Puedes combinar pulsaciones arriba o abajo, izquierda o derecha para seleccionar todo aquello que quieras.
- Shift + Inicio o Fin: Mueve el cursor al inicio o fin de la página y selecciona todo el texto por el que pasa.
- Tecla Shift + RePag o AvPag: Mueve el cursor al extremo superior o inferior de la pantalla visible seleccionando el texto.
- Tecla Ctrl + Shift + Inicio o Fin: Mueve el cursor al extremo superior o inferior del texto seleccionándolo a su vez.

BLOQUE II

TEMA 04

4. SISTEMAS OPERATIVOS PARA DISPOSITIVOS MÓVILES

Un sistema operativo móvil o SO móvil es un conjunto de programas de bajo nivel que permite la abstracción de las peculiaridades del hardware específico del teléfono móvil y, provee servicios a las aplicaciones móviles, que se ejecutan sobre él. Al igual que los PC que utilizan Windows, Linux o Mac OS, los dispositivos móviles tienen sus sistemas operativos como Android, iOS, entre otros. Los sistemas operativos móviles son mucho más simples y están más orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos.

Algunos de los sistemas operativos utilizados en los dispositivos móviles están basados en el modelo de capas.

Capas de un sistema operativo móvil

Núcleo (Kernel) ≡ El núcleo o kernel proporciona el acceso a los distintos elementos del hardware del dispositivo. Ofrece distintos servicios a las superiores como son los controladores o drivers para el hardware, la gestión de procesos, el sistema de archivos y el acceso y gestión de la memoria.

Middleware ≡ El middleware es el conjunto de módulos que hacen posible la propia existencia de aplicaciones para móviles. Es totalmente transparente para el usuario y ofrece servicios claves como el motor de mensajería y comunicaciones, códecs multimedia, intérpretes de páginas web, gestión del dispositivo y seguridad.

Entorno de ejecución de aplicaciones ≡ El entorno de ejecución de aplicaciones consiste en un gestor de aplicaciones y un conjunto de interfaces programables abiertas y programables por parte de los desarrolladores para la creación de software.

Interfaz de usuario ≡ Las interfaces de usuario facilitan la interacción con el usuario y el diseño de la presentación visual de la aplicación. Los servicios que incluye son el de componentes gráficos (botones, pantallas, listas, etc.) y el del marco de interacción.

4.1 RELACIÓN DE SISTEMAS OPERATIVOS PARA DISPOSITIVOS MÓVILES

Los más conocidos son obviamente Android e iOS, pero hay en realidad una larga lista de sistemas operativos para dispositivos móviles. No se pretende estudiar todos ellos, dado que no es en absoluto necesario, pero si se debe conocer su existencia.

LOS MÁS CONOCIDOS ≡ Android, iOS, Windows Phone, BlackBerry OS, Symbian OS, FireFox OS

SO IMPORTANTES BASADOS EN LINUX ≡ Chrome OS, Ubuntu Touch, Tizen, KaiOS, Sailfish OS, PostMarket, Lite OS, WebOS, LuneOS, PureOS,

SO IMPORTANTES BASADOS EN ANDROID ≡ ZenUI, ZUI, Replicant OS, realme UI, OxygenOS, LineageOS, Indus OS, GrapheneOS

Se puede ver una lista aún más completa en este enlace:

https://en.wikipedia.org/wiki/Mobile_operating_system#Chrome_OS

BLOQUE II

TEMA 04

4.2 SISTEMA OPERATIVO ANDROID

Android es un sistema operativo basado en Linux diseñado para dispositivos móviles, cuya arquitectura modular combina componentes nativos, máquinas virtuales, bibliotecas y capas de abstracción para ofrecer rendimiento, seguridad y escalabilidad.

Características

- Código abierto.
- Núcleo basado en el Kernel de Linux.
- Adaptable a muchas pantallas y resoluciones.
- Utiliza SQLite para el almacenamiento de datos.
- Ofrece diferentes formas de mensajería.
- Navegador web basado en WebKit incluido.
- Soporte de Java y muchos formatos multimedia.
- Soporte de HTML, HTML5, Adobe Flash Player, etc.
- Incluye un emulador de dispositivos, herramientas para depuración de memoria y análisis del rendimiento del software.
- Catálogo de aplicaciones gratuitas o pagas en el que pueden ser descargadas e instaladas (Google Play).
- Bluetooth.
- Google Talk desde su versión HoneyComb, para realizar videollamadas.
- Multitarea real de aplicaciones.

Arquitectura técnica de Android

La arquitectura de Android se organiza en capas jerárquicas, desde el hardware hasta las aplicaciones:

Linux Kernel ≡ Base del sistema operativo: gestiona memoria, procesos, seguridad, red y controladores de hardware. Android usa un kernel modificado de Linux con módulos específicos como Binder IPC, wakelocks, y Ashmem (memoria compartida).

Hardware Abstraction Layer (HAL) ≡ Interfaz entre el kernel y las bibliotecas nativas. Permite que Android funcione en distintos dispositivos sin modificar el sistema base. Ejemplo: camera HAL, audio HAL, sensors HAL.

Bibliotecas nativas (C/C++) ≡ libc(biblioteca estándar de C), OpenGL, WebKit, SQLite, etc.

Android Runtime (ART) ≡ Máquina virtual. Sustituyó a Dalvik desde Android 5.0. Ejecuta apps en formato DEX (Dalvik Executable). Usa compilación AOT (Ahead-of-Time) para mejorar rendimiento y reducir consumo de batería. Cada app corre en su propia instancia de ART, aislada por sandboxing.

Framework de aplicaciones ≡ API de alto nivel para desarrolladores. Permite que las apps interactúen con el sistema sin acceder directamente al hardware.

Aplicaciones del sistema y del usuario ≡ Apps como Teléfono, Mensajes, Configuración, y las que instala el usuario. Escritas en Java/Kotlin y compiladas a DEX para ejecutarse en ART.

BLOQUE II

TEMA 04

Máquinas virtuales y ejecución

- Dalvik VM (hasta Android 4.4): máquina virtual basada en registros, optimizada para dispositivos con poca memoria.
- ART (Android Runtime): Desde Android 5.0. Reemplazo moderno que compila apps al instalarse, mejora el rendimiento y reduce el uso de CPU en tiempo de ejecución.

Seguridad y aislamiento

- Cada app se ejecuta en su propio UID (User ID) y espacio de memoria.
- Android usa sandboxing, SELinux, y verificación de permisos para proteger el sistema.
- El sistema de permisos está basado en manifiestos declarativos y autorización en tiempo de ejecución desde Android 6.0.

Hardware

Android es compatible principalmente con procesadores basados en **arquitectura ARM**, aunque también admite x86 y, en menor medida, MIPS (en desuso actualmente). La arquitectura determina cómo se compilan y ejecutan las apps, y afecta directamente al rendimiento, consumo energético y compatibilidad.

Desde 2012, empezaron a aparecer dispositivos Android con procesadores Intel, incluidos teléfonos y tabletas. Mientras obtenía soporte para plataformas de 64 bits, primero se hizo Android para ejecutarse en x86 de 64 bits y luego en ARM64. Desde Android 5.0 "Lollipop", las variantes de 64 bits de todas las plataformas son compatibles además de las variantes de 32 bits.

Arquitectura	Descripción	Uso típico
ARM (armeabi, armeabi-v7a)	Arquitectura de 32 bits, eficiente en consumo. Muy extendida en móviles.	Gama baja/media
ARM64 (arm64-v8a)	Versión de 64 bits con soporte para AArch32 y AArch64.	Gama media/alta
x86	Arquitectura de Intel de 32 bits. Menos común, usada en algunas tablets y emuladores.	Tablets, emuladores
x86_64	Versión de 64 bits de x86. Compatible con apps x86 y x86_64.	Tablets, algunos portátiles
MIPS / MIPS64	Arquitectura experimental, casi en desuso en Android moderno.	Dispositivos embebidos (muy raro)

BLOQUE II

TEMA 04

4.2.1 VERSIONES DEL SISTEMA OPERATIVO ANDROID

Las versiones de Android recibían hasta la versión 9, y en inglés, el nombre de diferentes postres o dulces. En cada versión el postre o dulce elegido empieza por una letra distinta, conforme a un orden alfabético:

Nombre código	Número de versión	Nivel de API
Apple Pie ⁵⁵	1.0	1
Banana Bread ⁵⁵	1.1	2
Cupcake	1.5	3
Donut	1.6	4
Eclair	2.0 – 2.1	5 – 7
Froyo	2.2 – 2.2.3	8
Gingerbread	2.3 – 2.3.7	9 – 10
Honeycomb ⁵⁶	3.0 – 3.2.6	11 – 13
Ice Cream Sandwich	4.0 – 4.0.5	14 – 15
Jelly Bean	4.1 – 4.3.1	16 – 18
KitKat	4.4 – 4.4.4	19 – 20
Lollipop	5.0 – 5.1.1	21 – 22
Marshmallow	6.0 – 6.0.1	23
Nougat	7.0 – 7.1.2	24 – 25
Oreo	8.0 – 8.1	26 – 27
Pie	9.0	28
10	10.0	29
11	11.0	30

Android 11	—	2020	Burbujas de chat, permisos únicos.
Android 12	—	2021	Material You, privacidad mejorada.
Android 13	—	2022	Personalización por app, seguridad.
Android 14	—	2023	Optimización de batería, accesibilidad.
Android 15	—	2024	Mejoras en privacidad, rendimiento.
Android 16	—	2025	Integración con IA, multitarea avanzada

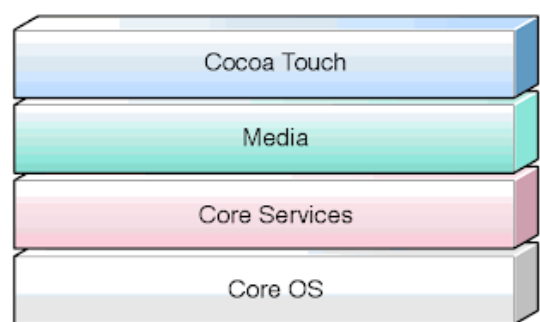
BLOQUE II**TEMA 04****4.3 SISTEMA OPERATIVO iOS**

iOS es un sistema operativo móvil de la multinacional Apple Inc. Originalmente desarrollado para el iPhone (iPhone OS), después se ha usado en dispositivos como el iPod touch y el iPad. Apple no permite la instalación de iOS en hardware de terceros. Desde iOS 5, las actualizaciones son **OTA (Over The Air)**.

iOS se deriva de macOS, que a su vez está basado en **Darwin BSD**, con núcleo XNU híbrido, y por lo tanto es un sistema operativo **Tipo Unix**, diseñado para dispositivos Apple con **procesadores ARM de 64 bits**.

Cuenta con **cuatro capas de abstracción**: la capa del núcleo del sistema operativo, la capa de "Servicios Principales", la capa de "Medios" y la capa de "Cocoa Touch".

- **Cocoa Touch** ≡ Cocoa Touch es la capa más importante para el desarrollo de aplicaciones iOS. Posee un conjunto de Frameworks que proporciona el API de Cocoa para desarrollar aplicaciones.



Se podría decir que Cocoa Touch proviene de Cocoa, la API ya existente en la plataforma MAC. Esta capa está formada por dos Frameworks fundamentales:

- UIKit: contiene todas las clases que se necesitan para el desarrollo de una interfaz de usuario
- Foundation Framework: define las clases básicas, acceso y manejo de objetos, servicios del sistema operativo
- **Media** ≡ Provee los servicios de gráficos y multimedia a la capa superior. Manejo de audio, vídeo, gráficos y animaciones. Incluye AVFoundation, Core Animation.
- **Core Services** ≡ Contiene los servicios fundamentales del sistema que usan todas las aplicaciones. APIs fundamentales como Foundation, Core Data, SQLite, y servicios de red
- **Core OS** ≡ Contiene las características de bajo nivel, base del sistema: Kernel, ficheros del sistema, manejo de memoria, seguridad, drivers del dispositivo, administración de energía, etc.

Seguridad y aislamiento

- iOS implementa una arquitectura de seguridad robusta:
- Sandboxing por aplicación: cada app corre en su propio entorno aislado.
- Control de permisos granular: acceso a cámara, micrófono, ubicación, etc.
- Secure Enclave: coprocesador dedicado para cifrado, Face ID, Touch ID.
- Actualizaciones OTA firmadas: solo Apple puede distribuir firmware.
- Protección de memoria: ASLR, DEP, y códigos firmados para evitar ejecución arbitraria.

BLOQUE II

TEMA 04

Componentes clave

- SpringBoard: gestor de escritorio e interfaz principal.
- Launchd: sistema de inicio y gestión de servicios.
- MobileGestalt: framework para obtener información del dispositivo.
- BackBoard: gestiona eventos táctiles y de hardware.

4.3.1 Jailbreak

El jailbreak es un proceso mediante el cual se eliminan las restricciones impuestas por Apple en el sistema operativo iOS, permitiendo acceso root al sistema, archivos y la instalación de software no autorizado por la App Store. Un dispositivo con jailbreak puede seguir usando la App Store, iTunes y las demás funciones normales, como por ejemplo realizar llamadas.

- Tethered jailbreak \equiv requiere que el dispositivo esté conectado a un ordenador cada vez que se inicie el sistema
- Untethered jailbreak \equiv permite al dispositivo iniciar sin ninguna asistencia adicional.
- Semitethered jailbreak \equiv consiste en ejecutar el jailbreak a través de una aplicación porque al reiniciar el sistema se elimina el jailbreak.

4.3.2 Tecnologías no admitidas

Resumen de las tecnologías no admitidas o restringidas por iOS en su versión actual:

Tecnología	Motivo de exclusión
Adobe Flash	Eliminado por razones de seguridad, rendimiento y obsolescencia.
Java Applets	No se permite ejecución de código Java embebido en navegador.
Apps de 32 bits	Desde iOS 11, solo se admiten apps de 64 bits.
Acceso directo al sistema de archivos	iOS no permite acceso al sistema de archivos completo por parte del usuario o apps.
Drivers de terceros	No se pueden instalar controladores personalizados como en macOS o Windows.
Ejecutables arbitrarios	No se permite ejecutar binarios fuera del entorno sandbox de apps.

BLOQUE II

TEMA 04

Restricciones de seguridad y arquitectura

- **No hay acceso root:** incluso para desarrolladores, el sistema está completamente cerrado.
- Sin emulación de sistemas operativos: no se permite virtualizar ni emular otros entornos.
- Sin compilación en tiempo real: los lenguajes como Python, Java o C no pueden compilarse directamente en el dispositivo sin usar apps específicas y sandboxed.

4.3.3 iOS 26. (última versión al momento de hacer este tema)

La última versión de iOS es **iOS 26**, lanzada oficialmente en septiembre de 2025. Representa el mayor rediseño del sistema operativo del iPhone en más de una década.

Novedades clave de iOS 26

- Nuevo lenguaje visual "**Liquid Glass**": una interfaz más fluida, translúcida y dinámica que redefine la experiencia visual.
- Mejoras en privacidad y seguridad: nuevas funciones para controlar el acceso a datos sensibles y mayor transparencia en el uso de sensores.
- Funciones inteligentes: integración más profunda de IA (**Apple Intelligence**) para sugerencias contextuales, escritura predictiva y automatización de tareas.
- Rediseño de apps nativas: como Mensajes, Fotos y CarPlay, con nuevas capacidades y estética unificada.
- Compatibilidad: disponible para una amplia gama de iPhones, aunque algunos modelos antiguos han quedado fuera del soporte

Criterios técnicos de compatibilidad

Requisito mínimo	Justificación técnica
Chip A13 Bionic o superior	Necesario para ejecutar el nuevo entorno visual "Liquid Glass" y funciones de IA
4 GB de RAM	Requisito para multitarea avanzada y procesamiento gráfico fluido
Neural Engine de 3.ª generación	Para funciones de privacidad, reconocimiento facial, y automatización contextual

Funciones eliminadas o limitadas

- Soporte para chips A12 o anteriores
- Widgets clásicos sin animación
- Compatibilidad con apps de 32 bits

Aunque algunos modelos antiguos pueden instalar iOS 26, no reciben todas las funciones, especialmente las relacionadas con Apple Intelligence.

BLOQUE II

TEMA 04

Evolución reciente de versiones de iOS

Apple ha alineado la numeración de iOS con macOS y watchOS, saltando directamente a la versión 26. La versión anterior a iOS 26 fue iOS 17, lanzada oficialmente en septiembre de 2023. Apple decidió saltar directamente de iOS 17 a iOS 26 en 2025 como parte de una estrategia de alineación de versiones entre sus sistemas operativos:

Versión	Año de lanzamiento	Notas clave
iOS 17	Septiembre 2023	Mejoras en Mensajes, AirDrop, widgets interactivos, y modo StandBy.
—	2024	Apple no lanzó iOS 18–25 como versiones públicas.
iOS 26	Septiembre 2025	Rediseño visual completo, Apple Intelligence, nueva numeración alineada con macOS y watchOS.