

Arquitectura de computadoras y sistemas operativos

Cómo interactúan el hardware, el software y los datos digitales

Este curso explorará los principios fundamentales que rigen cómo se construyen y cómo operan las computadoras. Profundizaremos en el funcionamiento interno de los componentes de hardware, desde los procesadores hasta la memoria, comprendiendo su diseño y función.

Además, examinaremos el papel fundamental de los sistemas operativos. Estas capas de software esenciales administran los recursos de hardware, ejecutan aplicaciones y brindan la interfaz que permite a los usuarios interactuar con la computadora sin problemas.

Al comprender tanto la arquitectura de la computadora como los sistemas operativos, obtendrá una visión integral de la intrincada relación entre los componentes físicos y los programas que les dan vida, lo que en última instancia permite el procesamiento de datos digitales que sustentan toda la informática moderna.

Objetivos del curso

Al final de esta presentación, podrá:

1 Comprender la arquitectura de la computadora y sus componentes básicos

Identificar los elementos de hardware fundamentales que componen las computadoras modernas y cómo funcionan en conjunto.

2 Analizar el papel de los sistemas operativos y su interacción con el hardware

Reconocer cómo los sistemas operativos cierran la brecha entre el hardware y las aplicaciones de usuario, gestionando eficazmente los recursos del sistema.

3 Interpretar cómo se representan y procesan los datos en una computadora

Comprender los fundamentos binarios de la informática y cómo la información fluye a través de los sistemas informáticos.

4 Explicar la gestión y la jerarquía de la memoria

Describir cómo se organiza, gestiona y optimiza la memoria de la computadora para diferentes tipos de acceso a los datos.

5 Discutir la gestión de procesos y los principios de multitarea

Comprender los conceptos de procesos, hilos y cómo los sistemas operativos permiten que varias tareas se ejecuten simultáneamente.

6 Evaluar las consideraciones de seguridad en los sistemas informáticos modernos

Identificar los desafíos de seguridad comunes y los mecanismos fundamentales empleados por el hardware y el software para proteger los datos y los sistemas.

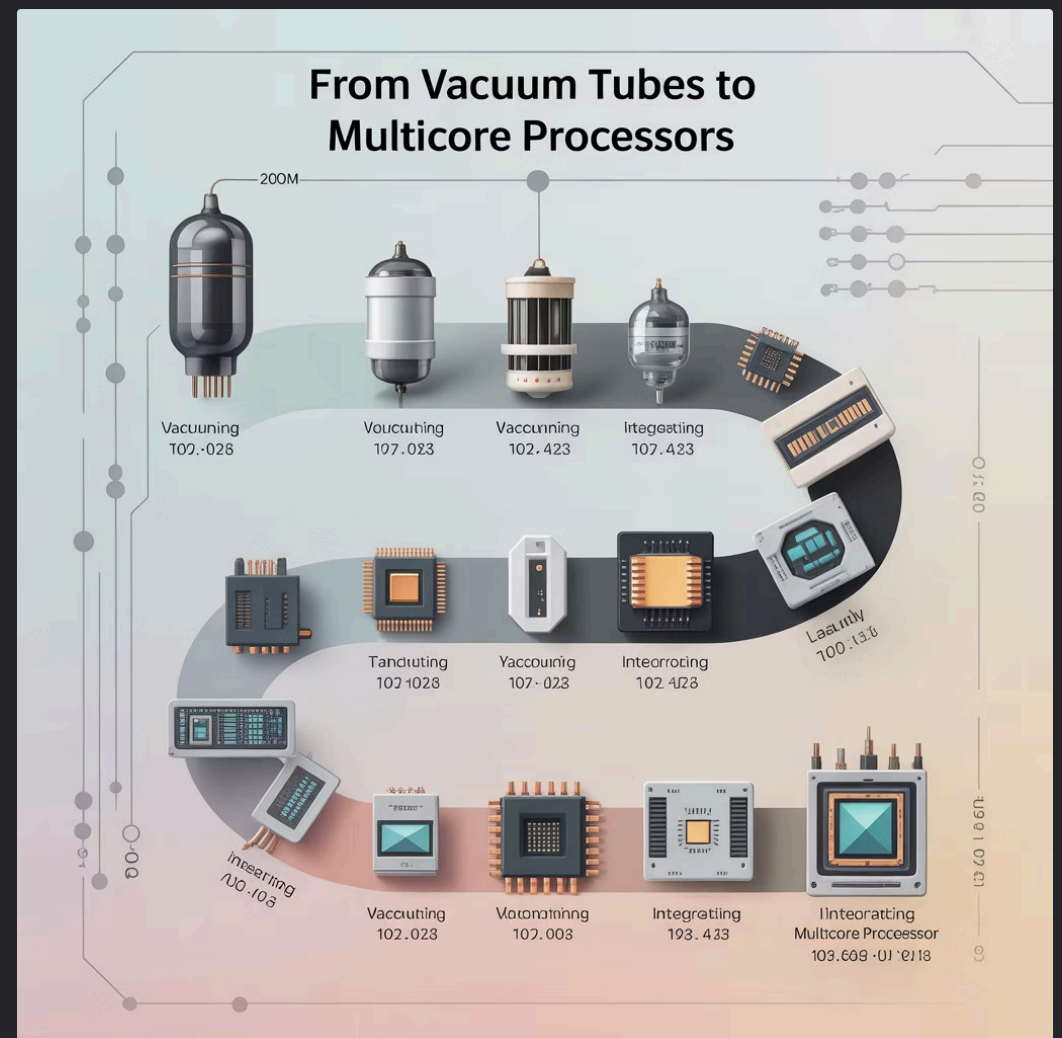
¿Qué es la arquitectura de computadoras?

La arquitectura de computadoras se refiere al conjunto fundamental de reglas, métodos y estructuras que definen el funcionamiento, la funcionalidad y la organización de los sistemas informáticos. Actúa como un plano, guiando el desarrollo de los componentes de hardware para garantizar que trabajen juntos de forma armoniosa y eficiente.

Los aspectos clave que abarca la arquitectura de computadoras incluyen:

- **El diseño y la organización de los componentes de hardware:** Esto implica definir las partes específicas de un sistema informático, como la Unidad Central de Procesamiento (CPU), varios tipos de memoria (RAM, cachés), dispositivos de entrada/salida (E/S) y unidades de almacenamiento permanente, junto con su disposición estructural.
- **Cómo interactúan estos componentes entre sí:** Especifica las vías de comunicación (buses) y los protocolos que permiten que las diferentes partes del sistema intercambien datos y señales de control sin problemas, garantizando un funcionamiento fluido y coordinado. Esto incluye la comprensión del flujo de datos y los mecanismos de control.
- **El conjunto de instrucciones que dicta las operaciones (ISA):** Esta es una interfaz crítica entre el hardware y el software, que define las operaciones básicas que un procesador puede realizar y cómo se codifican estas operaciones. Esencialmente, determina el lenguaje nativo que la CPU entiende y ejecuta.
- **La filosofía general de diseño del sistema:** Esto considera los objetivos generales y las compensaciones en el diseño de una computadora, como la optimización del rendimiento, el consumo de energía, la rentabilidad, la fiabilidad y la seguridad, en función de sus aplicaciones previstas y las necesidades del usuario.

Esencialmente, la arquitectura de computadoras tiene como objetivo optimizar el diseño y la interconexión de los componentes básicos de hardware para crear una máquina informática robusta y eficiente capaz de ejecutar las instrucciones de software de forma eficaz.



La evolución de la arquitectura de computadoras es un viaje fascinante que es paralelo a los importantes avances en la ciencia de los materiales y la electrónica. Ha progresado a través de distintas generaciones, cada una marcada por cambios tecnológicos innovadores.

Las primeras computadoras, por ejemplo, dependían de tubos de vacío voluminosos y de gran consumo de energía, lo que limitaba su escala y velocidad. La invención del transistor trajo un cambio revolucionario, lo que llevó a sistemas más pequeños, rápidos y fiables. A esto le siguió rápidamente el desarrollo de circuitos integrados, que permitieron empaquetar densamente millones de transistores en un solo chip de silicio, allanando el camino para la creación de microprocesadores y el auge de la informática personal.

En tiempos más recientes, la atención se ha desplazado hacia los complejos sistemas multinúcleo, donde se integran múltiples unidades de procesamiento independientes en un solo chip. Este paradigma permite el procesamiento paralelo, lo que mejora significativamente la potencia computacional para las aplicaciones exigentes y supera los límites de lo que las computadoras pueden lograr.

Componentes principales de una computadora

CPU (Unidad Central de Procesamiento)

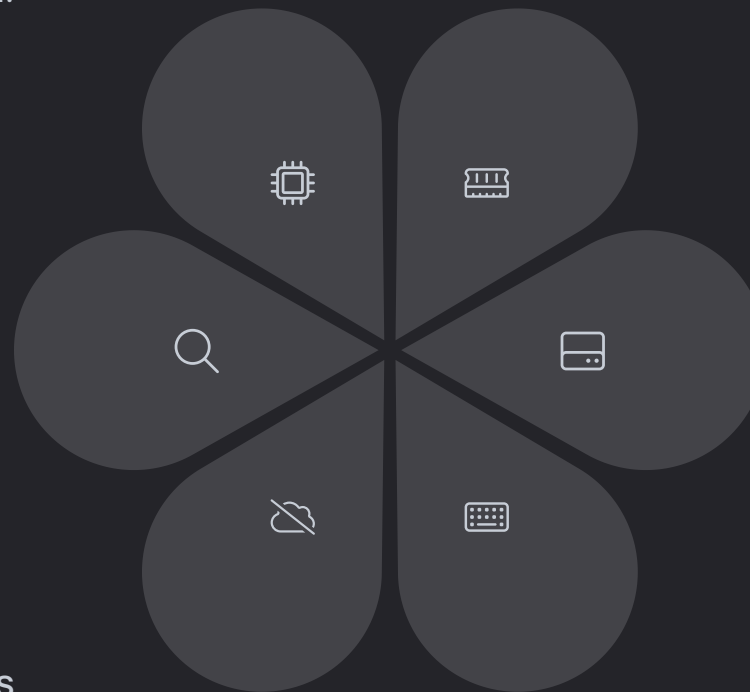
A menudo denominada el "cerebro" de la computadora, la CPU ejecuta instrucciones, realiza operaciones aritméticas y lógicas y gestiona el flujo general de datos. Su rendimiento es crucial para la velocidad y la eficiencia del sistema.

Placa base

La placa base es la placa de circuito principal que conecta todos los componentes de la computadora, incluida la CPU, la RAM y las tarjetas de expansión. Proporciona las conexiones eléctricas y las vías para la comunicación de datos, actuando como el centro del sistema.

Buses

Los buses son vías de comunicación que transfieren datos entre varios componentes dentro del sistema informático. Incluyen buses de datos para transportar información, buses de direcciones para especificar ubicaciones de memoria y buses de control para administrar operaciones.



Memoria principal (RAM)

La RAM (memoria de acceso aleatorio) es un tipo de almacenamiento volátil y temporal que contiene datos y programas que la CPU está utilizando activamente. Su velocidad permite un acceso rápido a la información, lo cual es vital para la multitarea y el rendimiento de las aplicaciones.

Almacenamiento secundario

Estos son dispositivos de almacenamiento permanentes no volátiles, como las unidades de disco duro (HDD) y las unidades de estado sólido (SSD), que conservan los datos incluso cuando la computadora está apagada. Se utilizan para el almacenamiento a largo plazo del sistema operativo, las aplicaciones y los archivos de usuario.

Dispositivos de entrada/salida

Los dispositivos de entrada/salida (E/S) facilitan la comunicación entre la computadora y el mundo externo. Los dispositivos de entrada, como los teclados y los ratones, permiten a los usuarios proporcionar datos, mientras que los dispositivos de salida, como los monitores y las impresoras, muestran o producen resultados.



**FETCH
EXECUTE
STORE**

El ciclo de instrucción

El ciclo de instrucción, también conocido como el ciclo de búsqueda-decodificación-ejecución, es el ciclo de operación fundamental de la unidad central de procesamiento (CPU) de una computadora. Representa la secuencia de pasos que sigue una CPU para procesar una instrucción desde el momento en que se recupera de la memoria hasta que se completa su ejecución y se almacenan los resultados. Este ciclo continuo forma la base de cómo una computadora opera y procesa la información.



Búsqueda

La CPU recupera una instrucción de la memoria utilizando la dirección que se encuentra actualmente en el Contador de Programa (PC). Esto implica colocar la dirección en el bus de direcciones y leer los datos de la instrucción desde el bus de datos en un registro interno de la CPU, normalmente el Registro de Instrucciones (IR).



Decodificar

Una vez obtenida, la instrucción es interpretada por la unidad de control (parte de la CPU) para determinar qué operación debe realizarse. La unidad de control descifra el código de operación (código de operación) e identifica cualquier operando (datos o direcciones de memoria) requerido para la instrucción. Esta etapa prepara las vías necesarias para el siguiente paso.



Ejecutar

Durante la etapa de ejecución, la CPU realiza la operación especificada por la instrucción. Esto podría implicar el uso de la Unidad Aritmético Lógica (ALU) para cálculos, el movimiento de datos entre registros o la interacción con dispositivos de entrada/salida. Las acciones específicas dependen completamente de la instrucción decodificada.



Almacenar

Después de la ejecución, se almacenan los resultados de la operación. Esto puede implicar escribir datos de nuevo en los registros dentro de la CPU, actualizar los valores en la memoria principal o enviar datos a un dispositivo de salida. Finalmente, el Contador de Programa (PC) se actualiza para apuntar a la dirección de la siguiente instrucción, preparando la CPU para el ciclo subsiguiente.

Administración de la memoria

Memoria primaria vs. secundaria

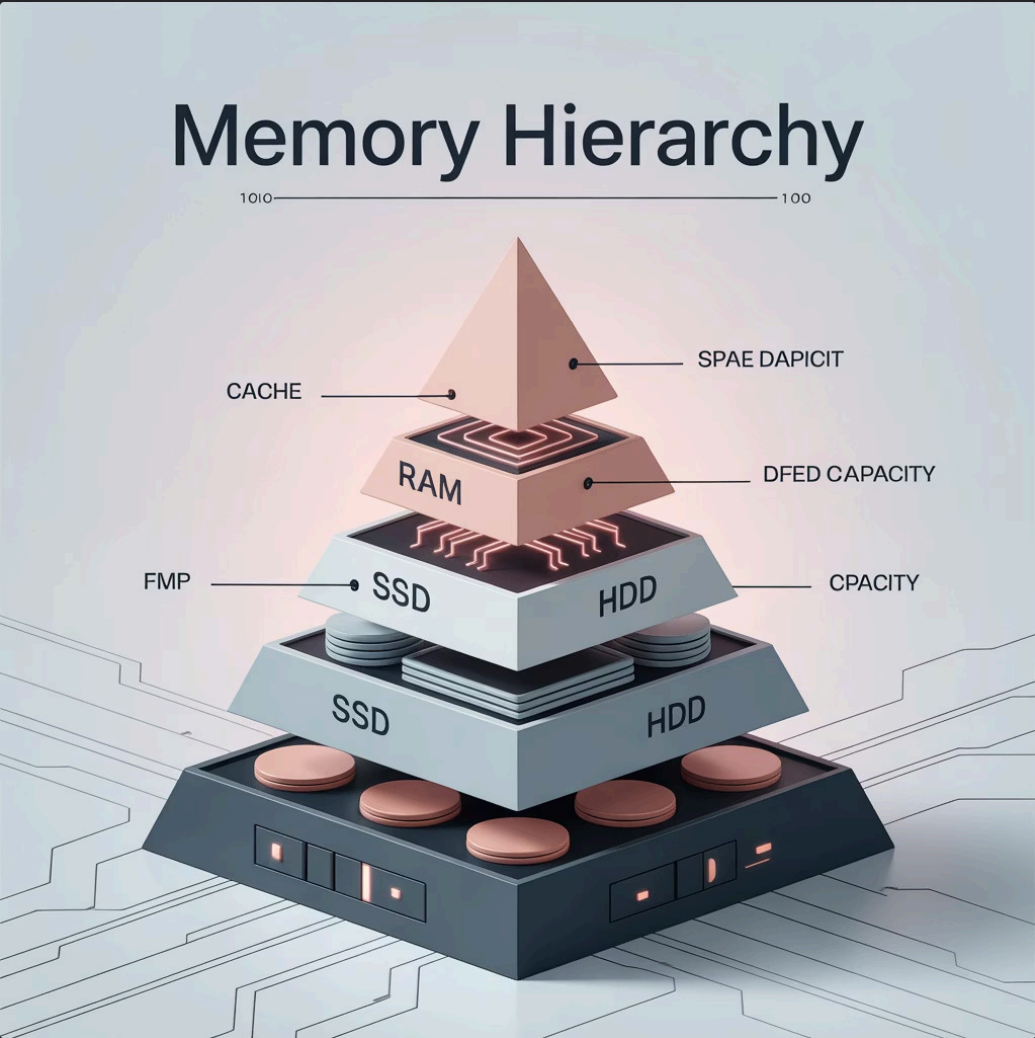
La memoria primaria, comúnmente conocida como RAM (Memoria de Acceso Aleatorio), se caracteriza por su volatilidad, lo que significa que pierde su contenido cuando se interrumpe la alimentación. Ofrece velocidades de acceso extremadamente rápidas, lo que permite a la CPU acceder directamente a los datos y las instrucciones. La RAM es esencial para ejecutar programas activos y almacenar datos que se están utilizando actualmente. En contraste, la memoria secundaria, como las unidades de estado sólido (SSD) y las unidades de disco duro (HDD), no es volátil y conserva los datos incluso sin alimentación. Si bien es significativamente más lenta que la RAM en términos de tiempo de acceso, el almacenamiento secundario ofrece capacidades mucho mayores a un costo menor por byte, lo que lo hace ideal para el almacenamiento de datos a largo plazo y los archivos del sistema.

Espacio de direcciones

El espacio de direcciones se refiere al rango completo de ubicaciones de memoria a las que puede acceder un programa o proceso. A cada ubicación se le asigna una dirección binaria única. Es importante destacar que se hace una distinción entre las direcciones lógicas (o virtuales), que son direcciones generadas por la CPU desde dentro de un programa, y las direcciones físicas, que corresponden a las ubicaciones de hardware reales en la memoria principal. El sistema operativo, junto con la Unidad de Administración de Memoria (MMU), es responsable de traducir estas direcciones lógicas en físicas, lo que garantiza que los programas puedan ejecutarse de manera eficiente y segura sin interactuar directamente con el hardware de la memoria física.

Memoria virtual

La memoria virtual es una técnica fundamental de administración de memoria que permite a una computadora compensar la escasez de memoria física transfiriendo temporalmente datos de la RAM al almacenamiento en disco. Crea la ilusión de que un proceso tiene un bloque contiguo de memoria, incluso si está fragmentado o almacenado parcialmente en el disco. Esta técnica permite que un sistema ejecute más programas simultáneamente de los que cabrían en la RAM física, mejora la protección de la memoria al aislar los procesos y simplifica la programación al permitir que las aplicaciones direccionen un espacio de memoria grande y uniforme sin preocuparse por las limitaciones físicas.



El sistema operativo desempeña un papel fundamental en la administración de la memoria de la computadora para garantizar un funcionamiento eficiente y seguro. Sus responsabilidades incluyen:

Asignación de memoria para programas: El sistema operativo asigna dinámicamente bloques de memoria a varios programas en ejecución y procesos del sistema a medida que lo requieren. Esto implica realizar un seguimiento de qué partes de la memoria están en uso y cuáles están libres, y garantizar que cada proceso reciba la cantidad necesaria de memoria sin superponerse ni causar conflictos con otros procesos.

Protección entre diferentes procesos: Una función clave de la administración de la memoria es proporcionar protección entre diferentes procesos. El sistema operativo garantiza que un programa no pueda acceder o modificar accidental o maliciosamente el espacio de memoria asignado a otro programa o al propio sistema operativo. Esto evita el acceso no autorizado, mantiene la estabilidad del sistema y mejora la seguridad.

Asignación de direcciones virtuales a físicas: Como parte de la memoria virtual, el sistema operativo, a menudo con la ayuda del hardware de la Unidad de Administración de Memoria (MMU), realiza la tarea crucial de asignar direcciones virtuales generadas por los procesos a direcciones físicas reales en la RAM. Esta traducción es vital para implementar la memoria virtual y el aislamiento de procesos, lo que permite que varios programas compartan la memoria física limitada sin interferir con los espacios de direcciones de los demás.

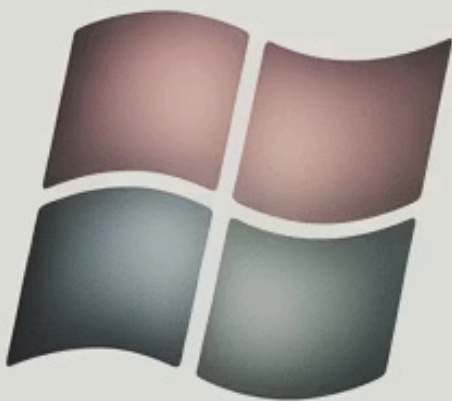
Intercambio de datos entre la RAM y el disco según sea necesario: Cuando la memoria total requerida por los programas activos excede la RAM física disponible, el sistema operativo emplea técnicas como el intercambio y la paginación. Mueve (intercambia) los datos y segmentos de programa que se utilizan con menos frecuencia de la RAM a un área designada en el disco duro (espacio de intercambio o archivo de página). Cuando estos segmentos se necesitan nuevamente, se vuelven a mover a la RAM. Este proceso, aunque más lento, permite que el sistema administre la memoria de manera eficiente y ejecute más aplicaciones de lo que permitiría la memoria física por sí sola.



dows



Lin



COS



And

¿Qué es un sistema operativo?

Un sistema operativo (SO) es el software fundamental que gestiona los recursos de hardware y software de la computadora. Proporciona un entorno estable y consistente para que se ejecuten las aplicaciones, sirviendo como puente entre las aplicaciones de usuario y los componentes de hardware sin procesar. Sin un sistema operativo, una computadora sería una colección de piezas desconectadas.



Administrador de recursos

Asigna y coordina los recursos de hardware, como el tiempo de CPU, la memoria y los dispositivos de E/S entre múltiples programas y usuarios.



Capa de abstracción de hardware

Proporciona interfaces estandarizadas que ocultan los detalles complejos del hardware de las aplicaciones y los usuarios, simplificando la programación.



Proveedor de servicios

Ofrece servicios básicos como administración de archivos, seguridad, redes e interfaces de usuario, esenciales para la interacción del usuario y la ejecución de aplicaciones.



Programador de procesos

Gestiona la ejecución de múltiples programas (procesos) simultáneamente, determinando qué proceso tiene acceso a la CPU y durante cuánto tiempo, garantizando una utilización eficiente de los recursos.



Interfaz de usuario

Proporciona un medio para que los usuarios interactúen con la computadora, principalmente a través de interfaces gráficas de usuario (GUI) o interfaces de línea de comandos (CLI), lo que permite tareas como iniciar aplicaciones y administrar archivos.

En esencia, el sistema operativo es el sistema nervioso central de una computadora, que orchestra todas las operaciones y garantiza que el software y el hardware funcionen juntos sin problemas. Es el primer programa que se carga cuando se inicia una computadora y se ejecuta continuamente hasta que se apaga la computadora.

Interacción entre el hardware y el sistema operativo

El sistema operativo actúa como un intermediario esencial, orquestando una comunicación y un control perfectos entre los componentes de hardware del ordenador y las aplicaciones de software que se ejecutan en él. Esta intrincada interacción es fundamental para la funcionalidad y el rendimiento de un ordenador, garantizando que los recursos se gestionen de forma eficiente y que las tareas se ejecuten sin problemas.

Gestión de procesos

Al crear, programar y finalizar programas, el sistema operativo gestiona meticulosamente la ejecución de las aplicaciones. Emplea sofisticados algoritmos de programación para decidir qué proceso tiene acceso a la CPU y durante cuánto tiempo, garantizando una asignación justa de los recursos y una multitarea eficiente. Esto implica un cambio de contexto frecuente entre los procesos para dar la ilusión de una ejecución simultánea.

Gestión de la memoria

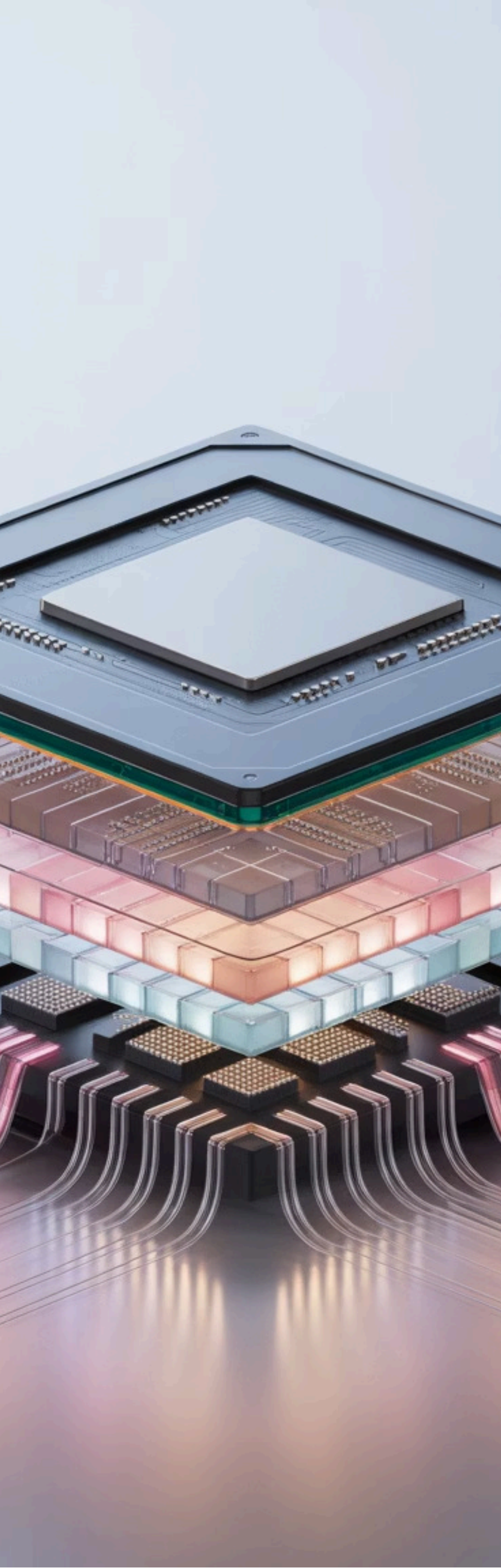
Al asignar RAM a varios procesos, el sistema operativo es crucial para una utilización eficiente de la memoria. Implementa la memoria virtual, lo que permite a los programas utilizar más memoria de la disponible físicamente intercambiando datos de forma transparente entre la RAM y los dispositivos de almacenamiento secundario, como los discos duros. Además, protege los distintos espacios de memoria, impidiendo que un programa interfiera o corrompa los datos de otro.

Gestión de archivos

Al organizar, almacenar y recuperar datos en varios dispositivos de almacenamiento, el sistema operativo proporciona un sistema de archivos estructurado. Esto permite a los usuarios y a las aplicaciones interactuar con los datos de forma abstracta, sin necesidad de conocer la ubicación física en el disco. También gestiona los permisos de acceso a los archivos, garantizando la seguridad de los datos, y gestiona operaciones como la creación, la eliminación, la lectura y la escritura de archivos.

Gestión de E/S

El control de todos los dispositivos de entrada/salida (E/S), desde teclados y ratones hasta tarjetas de red e impresoras, es una función central del sistema operativo. Utiliza controladores de dispositivo específicos para comunicarse con cada componente de hardware único y gestiona las interrupciones generadas por los dispositivos para responder a los eventos en tiempo real. El almacenamiento en búfer de los datos ayuda a gestionar las diferencias de velocidad entre la CPU y los dispositivos de E/S más lentos, mejorando la capacidad de respuesta y la eficiencia del sistema.



Tipos de hardware y términos clave

Procesadores

- **Núcleos:** Unidades de procesamiento independientes dentro de una CPU, cada una capaz de ejecutar instrucciones simultáneamente. Las CPU modernas a menudo tienen múltiples núcleos para manejar más tareas en paralelo.
- **Hilos:** Núcleos virtuales que comparten los recursos físicos de un solo núcleo. Permiten que un núcleo gestione múltiples secuencias de instrucciones simultáneamente, mejorando significativamente la eficiencia para muchas aplicaciones.
- **Arquitectura:** La filosofía de diseño fundamental detrás del conjunto de instrucciones y la estructura interna de una CPU (por ejemplo, CISC - Computadora con Conjunto de Instrucciones Complejas, RISC - Computadora con Conjunto de Instrucciones Reducidas). Esta elección impacta profundamente en cómo se procesan las instrucciones.

Reloj del sistema

- **Frecuencia medida en Hertz (GHz),** que representa miles de millones de ciclos por segundo. Una frecuencia de reloj más alta generalmente indica capacidades de procesamiento más rápidas.
- **Determina la velocidad de ejecución de la CPU y sincroniza todos los componentes dentro del sistema informático,** asegurando que las operaciones a través de diferentes elementos de hardware ocurran de manera coordinada.

Memoria caché

- **L1:** El nivel más pequeño y rápido, ubicado más cerca de los núcleos de la CPU. Almacena las instrucciones y los datos a los que el núcleo accede con mayor frecuencia para su recuperación inmediata, minimizando la latencia.
- **L2:** Más grande que L1 y ligeramente más lento, a menudo dedicado por núcleo o compartido por un pequeño grupo de núcleos. Sirve como un búfer secundario de alta velocidad para los datos que no se encuentran en la caché L1.
- **L3:** El más grande y lento de los niveles de caché, típicamente compartido entre todos los núcleos en el chip del procesador. Actúa como un grupo común de memoria rápida para reducir la dependencia de la RAM principal, que es mucho más lenta.

Buses

- **Bus de datos:** Transfiere los datos reales entre la CPU, la memoria y los dispositivos de entrada/salida. Su ancho (por ejemplo, 64 bits) dicta cuántos datos se pueden transferir en una sola operación.
- **Bus de direcciones:** Especifica las ubicaciones de memoria únicas donde los datos deben leerse o escribirse. El número de líneas en el bus de direcciones determina la cantidad máxima de memoria física a la que puede acceder la CPU.
- **Bus de control:** Transporta señales de comando y estado entre la CPU y otros componentes, coordinando operaciones como señales de lectura/escritura, solicitudes de interrupción y reconocimiento de transferencias de datos.

Representación de datos

Base binaria: el lenguaje digital

En esencia, cada pieza de información procesada por una computadora, desde un simple carácter de texto hasta un video complejo, se descompone en su forma más fundamental: dígitos binarios o bits. Este enfoque universal simplifica los datos complejos en un formato que las computadoras pueden entender y manipular fácilmente.

- Bit:** La unidad de datos más pequeña, que representa uno de dos estados distintos: 0 (apagado) o 1 (encendido). Estos estados se corresponden directamente con las señales eléctricas dentro de los circuitos de la computadora, lo que los hace ideales para el procesamiento automático.
- Byte:** Una agrupación fundamental de 8 bits. Con 8 bits, un solo byte puede representar 28 (256) valores diferentes, lo que lo convierte en una unidad versátil para almacenar caracteres individuales, números pequeños o códigos de comandos básicos.
- Palabra:** Una colección de bits que representa la unidad natural de datos que un procesador específico maneja a la vez. Los tamaños de palabra comunes son 32 bits o 64 bits, lo que influye directamente en las capacidades de procesamiento de la computadora y en cómo direcciona la memoria.

La dependencia de lo binario se basa en la fiabilidad y la simplicidad de representar dos estados eléctricos distintos (voltaje alto/voltaje bajo o presencia/ausencia de una señal), lo que garantiza un manejo de datos coherente y eficiente.

Codificación de texto: más allá de los caracteres simples

Para representar texto legible por humanos, las computadoras utilizan esquemas de codificación que asignan patrones binarios específicos a los caracteres, lo que permite que el texto se almacene, transmita y muestre digitalmente.

ASCII (Código estándar estadounidense para el intercambio de información): Uno de los estándares más antiguos y extendidos, ASCII utiliza 7 bits para representar 128 caracteres. Esto incluye el alfabeto inglés (tanto en mayúsculas como en minúsculas), números, signos de puntuación y caracteres de control básicos. Por ejemplo:

```
'A' = 01000001 (representación ASCII)
```

Unicode y UTF-8: A medida que la informática se globalizó, la limitación de ASCII a los caracteres principalmente en inglés se convirtió en un obstáculo importante. Unicode se desarrolló para abordar esto proporcionando un identificador numérico único para cada carácter en todos los sistemas de escritura, incluidos varios idiomas, símbolos y emojis. UTF-8 es una codificación de ancho variable para Unicode, diseñada para una transmisión y almacenamiento eficientes. Se ha convertido en la codificación de caracteres dominante para Internet y la mayoría del software moderno, ya que admite miles de millones de caracteres únicos y garantiza la compatibilidad universal del texto.

Sistemas numéricos: simplificación de binarios complejos

Si bien todas las operaciones informáticas se realizan utilizando binario, los humanos a menudo confían en otros sistemas numéricos por conveniencia y legibilidad, particularmente cuando trabajan con grandes secuencias binarias o direcciones de memoria.

Decimal (base 10)	Binario (base 2)	Hexadecimal (base 16)
0	0000	0
1	0001	1
10	1010	A
15	1111	F

Hexadecimal: Este sistema de base 16 utiliza los dígitos 0-9 y las letras A-F. Su principal ventaja es su relación directa con el binario: cada dígito hexadecimal representa exactamente cuatro bits. Esto hace que el hexadecimal sea una abreviatura muy eficiente y conveniente para representar secuencias binarias largas (por ejemplo, 11110000 en binario es simplemente F0 en hexadecimal). Se usa ampliamente en la programación de computadoras para direcciones de memoria, códigos de color (como #FF0000 para rojo) y para mostrar datos sin procesar, ya que es mucho más legible por humanos y menos propenso a errores que largas cadenas de 0 y 1.

Representación de otros medios: imágenes y audio

Más allá del texto y los números, todas las demás formas de medios (imágenes, audio y video) también se convierten en datos binarios para el procesamiento, almacenamiento y transmisión por computadora. Este proceso es fundamental para los medios digitales.

- Imágenes:** Las imágenes digitales se componen de una cuadrícula de pequeños elementos de imagen llamados píxeles. A cada píxel se le asigna un valor binario que representa su color e intensidad. Por ejemplo, en un modelo de color RGB (rojo, verde, azul), cada componente de color tiene su propia representación binaria, y la combinación de estos crea el color final del píxel.
- Audio:** Las ondas de sonido, que son de naturaleza analógica, se convierten en datos digitales a través de un proceso llamado muestreo. La amplitud (volumen) de la onda de sonido se mide a intervalos regulares, y cada medición se convierte en un número binario. Una frecuencia de muestreo y una profundidad de bits más altas dan como resultado una representación de audio digital más precisa y de mayor fidelidad.
- Video:** El video digital es esencialmente una secuencia rápida de imágenes fijas (fotogramas) combinadas con pistas de audio sincronizadas. Cada fotograma es una imagen digital individual codificada en binario, y el audio que lo acompaña se digitaliza por separado. Estos flujos binarios luego se entrelazan y comprimen para un almacenamiento y reproducción eficientes.

La conversión de cualquier información analógica (como ondas de luz para imágenes u ondas de sonido para audio) en datos binarios digitales se conoce como digitalización, un concepto fundamental que permite las vastas capacidades de la informática moderna y la multimedia.

Código máquina y lenguaje ensamblador

Código máquina

Instrucciones binarias ejecutadas directamente por la CPU. Diferente para cada arquitectura de procesador. Este código binario sin procesar manipula directamente los componentes de hardware de la CPU, lo que representa el nivel más bajo de programación. Es único para cada tipo de procesador específico, lo que hace que sea extremadamente difícil para los humanos escribirlo o leerlo directamente.

```
10110000 01100001
```

Lenguaje ensamblador

Representación simbólica legible por humanos del código máquina. El ensamblaje utiliza mnemónicos (como MOV, ADD, JMP) y etiquetas, lo que ofrece una ligera abstracción sobre el código máquina puro. A menudo se utiliza para tareas de rendimiento crítico, desarrollo de sistemas operativos o programación de sistemas de bajo nivel, ya que proporciona control directo sobre el hardware sin tener que lidiar con binarios puros.

```
MOV AL, 61h ; Mueve el valor hexadecimal 61 al registro AL
```



Proceso de traducción

Código de alto nivel (C, Java) → Compilador → Ensamblaje → Ensamblador → Código máquina

La CPU solo puede ejecutar código máquina. Todos los lenguajes de nivel superior deben traducirse a este formato binario. Esta traducción es un proceso esencial de varios pasos. Un compilador primero convierte el código fuente de alto nivel legible por humanos en lenguaje ensamblador. Posteriormente, un ensamblador traduce este código ensamblador en el código máquina binario preciso que la CPU puede comprender y ejecutar directamente. Este enfoque en capas permite a los desarrolladores escribir software complejo de manera más eficiente, al tiempo que garantiza un rendimiento óptimo en el hardware subyacente.

Interrupciones

¿Qué es una interrupción?

Una señal enviada a la CPU que detiene temporalmente la ejecución normal para manejar un evento de mayor prioridad. Es como si alguien te tocara el hombro mientras estás trabajando, exigiendo tu atención inmediata.

Las interrupciones son cruciales para permitir que la CPU gestione eficientemente las operaciones de E/S y gestione eventos asíncronos sin sondeos constantes. Aseguran la capacidad de respuesta y permiten el buen funcionamiento de la multitarea en los sistemas operativos modernos.

Tipos de interrupciones

- **Interrupciones de hardware:** Generadas por dispositivos externos (por ejemplo, al presionar una tecla, el movimiento del ratón, el vencimiento de un temporizador o la llegada de datos de una tarjeta de red). Señalan que un dispositivo necesita atención inmediata.
- **Interrupciones de software:** Solicitudes explícitas de un programa en ejecución al kernel del sistema operativo para servicios, como leer un archivo, asignar memoria o imprimir un documento (a menudo llamadas llamadas al sistema).
- **Excepciones:** Eventos inesperados o errores que ocurren dentro de la CPU durante la ejecución del programa, como intentar dividir por cero, acceder a una dirección de memoria no válida o una condición de desbordamiento/subdesbordamiento.

Manejo de interrupciones

Cuando ocurre una interrupción, la CPU sigue una secuencia precisa de pasos:

1. **La CPU guarda su estado actual:** La CPU pausa temporalmente su tarea actual y guarda información crucial como el contador de programa actual, los valores de los registros y los indicadores de estado en la pila. Esto preserva el contexto del proceso interrumpido.
2. **Salta a una dirección específica:** La CPU determina el tipo de interrupción y encuentra la dirección del controlador de interrupciones apropiado en una tabla predefinida (la Tabla de Vectores de Interrupción).
3. **Ejecuta la Rutina de Servicio de Interrupción (ISR):** La CPU luego salta y ejecuta la ISR, que es una pieza de código dedicada diseñada para procesar el evento específico (por ejemplo, leer la entrada del teclado, completar una operación de disco o manejar un error).
4. **Regresa a la tarea anterior:** Después de que la ISR completa su trabajo, la CPU restaura el estado guardado de la pila y reanuda la ejecución normal del programa original desde donde lo dejó.

Importancia de las interrupciones

Las interrupciones son fundamentales para el funcionamiento de cualquier sistema informático moderno. Permiten una utilización eficiente de los recursos al permitir que la CPU realice otras tareas mientras espera que se completen las operaciones de E/S. Sin interrupciones, la CPU tendría que sondear (verificar) constantemente cada dispositivo en busca de actividad, lo que provocaría ciclos desperdiciados, un rendimiento deficiente y la incapacidad de ejecutar múltiples programas simultáneamente. Son la columna vertebral de la multitarea y la capacidad de respuesta del sistema en tiempo real.

Cómo se conecta todo

Interacción del usuario

El usuario inicia una aplicación a través de una interfaz gráfica, por ejemplo, haciendo clic en un icono. Esta acción envía una solicitud o comando específico al sistema operativo, indicando la intención del usuario de ejecutar un programa. El SO luego se prepara para cumplir con esta solicitud.

Preparación del SO

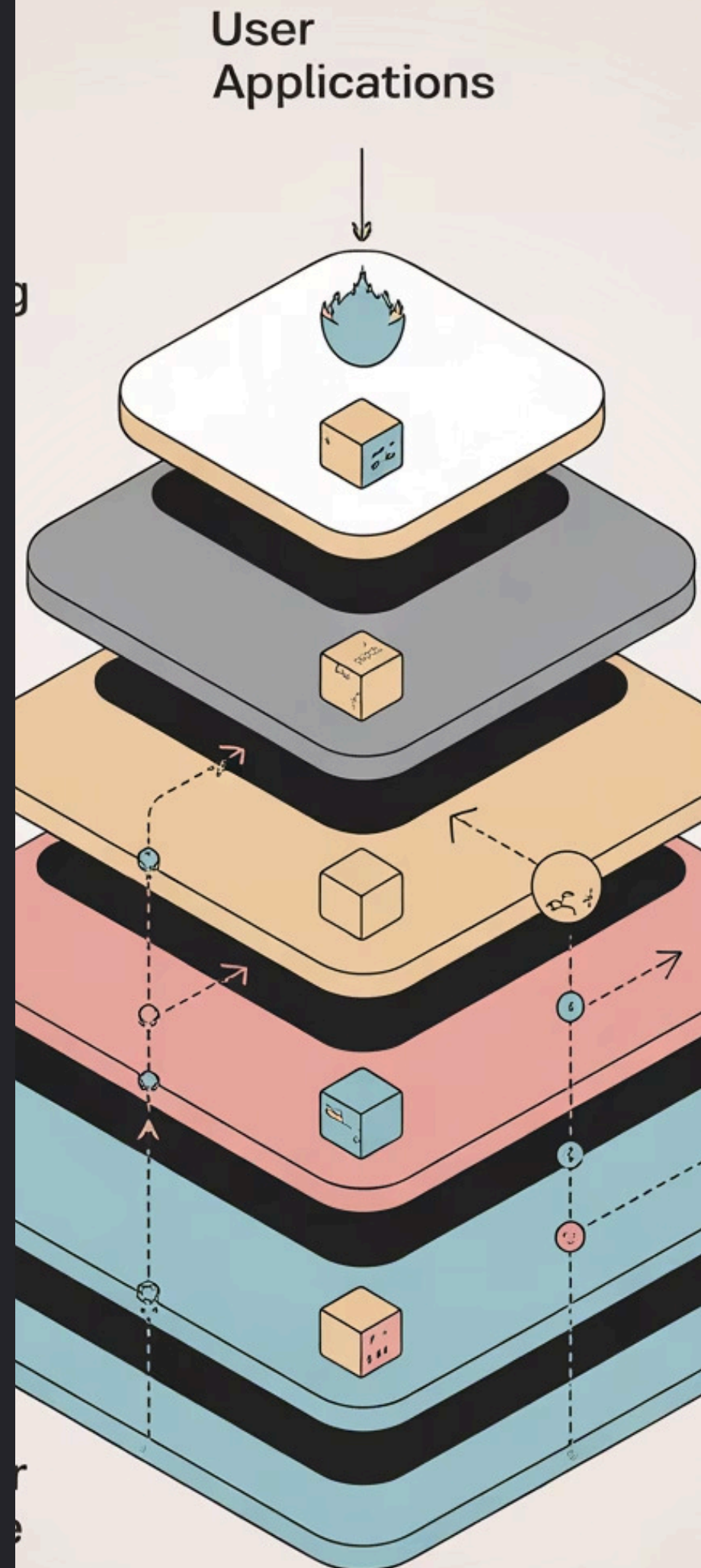
Al recibir la solicitud, el sistema operativo se hace cargo. Primero, asigna un bloque dedicado de memoria virtual para la aplicación, asegurándose de que tenga espacio para operar. Luego, el SO carga el código ejecutable del programa y los datos necesarios desde el almacenamiento secundario (como un disco duro) en la memoria de acceso aleatorio (RAM) más rápida de la computadora. Finalmente, crea un "proceso", que es una instancia del programa en ejecución, completa con su propio ID de proceso y recursos únicos.

Ejecución de la CPU

Una vez que el programa se carga y se crea un proceso, la CPU (Unidad Central de Procesamiento) comienza su función principal: ejecutar instrucciones. Realiza continuamente el ciclo de instrucción: busca instrucciones de código de máquina en la RAM, las decodifica para comprender el comando y luego ejecuta la operación (realiza cálculos, mueve datos, etc.). Este ciclo se repite millones de veces por segundo, impulsando la funcionalidad de la aplicación.

Coordinación de recursos

A lo largo de la ejecución de la aplicación, el sistema operativo permanece en control, coordinando constantemente los recursos. Maneja varios eventos, como interrupciones de dispositivos de hardware (como un clic del mouse o la llegada de datos de la red) o interrupciones de software (llamadas al sistema desde el propio programa), pausando y reanudando tareas según sea necesario. El SO también emplea algoritmos de programación para administrar múltiples procesos en ejecución, asegurando un acceso justo a la CPU y otros recursos. Esta supervisión continua garantiza el funcionamiento fluido y eficiente de todas las aplicaciones y del sistema en su conjunto.



Actividad de repaso

¿Verdadero o falso?

1. La CPU accede directamente a los datos del disco duro cuando se ejecuta un programa.

Falso. El sistema operativo carga primero los datos del almacenamiento en la RAM y, a continuación, la CPU accede a ellos.

2. La memoria virtual permite que un ordenador ejecute programas más grandes que su RAM física.

Verdadero. Utiliza el almacenamiento secundario como una extensión de la RAM.

3. Todos los sistemas operativos utilizan el mismo formato de código máquina.

Falso. El código máquina es específico de la arquitectura del procesador.

4. El ciclo de instrucción consta de: búsqueda, decodificación, ejecución y almacenamiento.

Verdadero. Estas cuatro fases representan el funcionamiento básico de la CPU.

5. Una interrupción es una señal que pausa la CPU para gestionar un evento de alta prioridad.

Verdadero. Las interrupciones permiten al sistema operativo gestionar las interacciones de hardware y las tareas críticas.

6. La RAM se considera memoria no volátil, lo que significa que conserva los datos cuando se apaga la alimentación.

Falso. La RAM es volátil; los datos se pierden cuando se corta la alimentación. La ROM o las SSD son no volátiles.

7. El kernel es el componente central de un sistema operativo.

Verdadero. Gestiona los recursos del sistema y proporciona servicios esenciales.

8. La gestión de procesos implica la creación, la programación y la finalización de los procesos.

Verdadero. El sistema operativo garantiza que varios programas puedan ejecutarse simultáneamente sin interferencias.

Pregunta de debate

¿Qué parte de un sistema informático parece más invisible pero esencial?

Si bien a menudo interactuamos con hardware y software visibles, muchas funciones críticas operan silenciosamente en segundo plano, lo que hace que nuestros sistemas sean realmente utilizables.

Considere:

- Administración de memoria
- Programación de procesos
- Manejo de interrupciones
- Sistemas de archivos
- Controladores de dispositivos

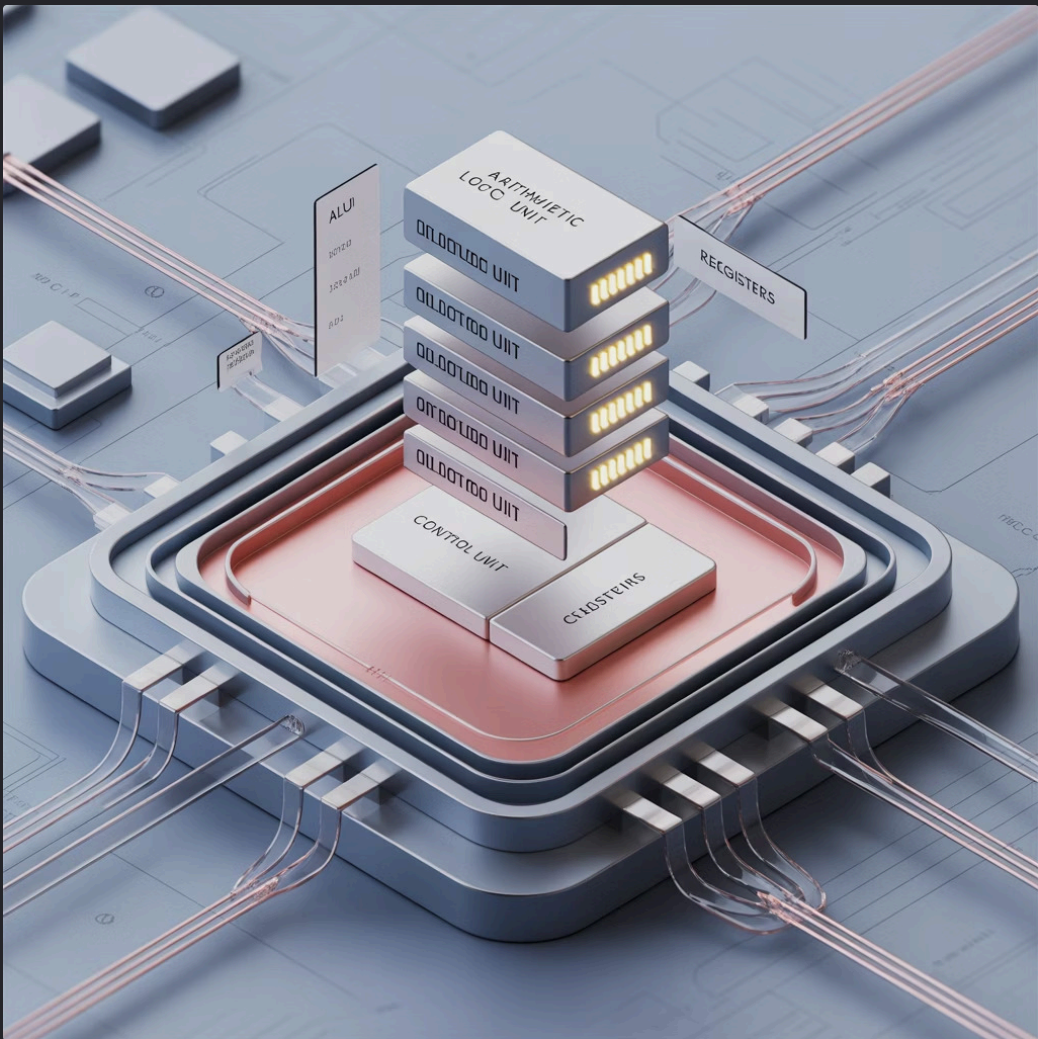


Considere el profundo impacto si este componente dejara de funcionar repentinamente o dejara de funcionar. ¿Cómo afectaría directamente tal avería a sus tareas diarias, desde navegar por la web hasta completar un trabajo importante?

La CPU en detalle

Componentes principales

- **Unidad de control (CU):** A menudo considerada el "cerebro" dentro de la CPU, la CU dirige y coordina todas las operaciones. Obtiene instrucciones de la memoria, las decodifica y luego gestiona el flujo de ejecución, enviando señales de control a otros componentes.
- **Unidad aritmético-lógica (ALU):** Este es el circuito digital que realiza todas las operaciones aritméticas (suma, resta, multiplicación, división) y operaciones lógicas (AND, OR, NOT, XOR, comparaciones). Es el componente central para la computación dentro de la CPU.
- **Registros:** Estas son pequeñas ubicaciones de almacenamiento extremadamente rápidas ubicadas directamente dentro de la CPU. Contienen datos en los que la CPU está trabajando actualmente, como operandos para cálculos, direcciones de memoria o resultados de instrucciones, proporcionando acceso inmediato a información crítica.
- **Caché:** Una pequeña cantidad de memoria de muy alta velocidad integrada en la CPU o cerca de ella. Almacena datos e instrucciones a los que se accede con frecuencia, lo que reduce la necesidad de acceder a la memoria principal (RAM) más lenta y acelera significativamente el procesamiento. Las CPU suelen tener múltiples niveles de caché (L1, L2, L3).



Las CPU modernas contienen miles de millones de transistores y pueden ejecutar miles de millones de instrucciones por segundo. El diseño arquitectónico determina significativamente el rendimiento, el consumo de energía y la compatibilidad con varios programas y hardware.

La innovación continua en el diseño de la CPU se centra en aumentar la potencia de procesamiento mientras se gestiona la disipación de calor y la eficiencia energética. Los avances en los procesos de fabricación permiten colocar más transistores en un solo chip, lo que conduce a una mayor complejidad y capacidades.

Más allá de estos componentes centrales, varios otros factores contribuyen al rendimiento y las capacidades generales de una CPU:



Velocidad de reloj

Medida en gigahercios (GHz), la velocidad de reloj indica cuántos ciclos de instrucción puede completar la CPU por segundo. Una velocidad de reloj más alta generalmente significa un procesamiento más rápido, aunque no es el único factor.



Núcleos

Un núcleo de CPU es una unidad de procesamiento que puede realizar tareas de forma independiente. Las CPU de varios núcleos (doble núcleo, cuatro núcleos, ocho núcleos, etc.) permiten el procesamiento paralelo, lo que mejora significativamente el rendimiento para la multitarea y las aplicaciones exigentes.



Hilos

Los hilos son secuencias de instrucciones que un programador puede gestionar de forma independiente. Muchas CPU modernas utilizan "hyper-threading" o "multi-threading simultáneo" para permitir que cada núcleo físico gestione múltiples hilos simultáneamente, lo que aumenta la eficiencia.



Arquitectura del conjunto de instrucciones (ISA)

La ISA define el conjunto de instrucciones que una CPU puede entender y ejecutar. Las ISA comunes incluyen x86 (utilizada en la mayoría de las computadoras de escritorio y portátiles) y ARM (predominante en dispositivos móviles).

Jerarquía de memoria



Registros

Los registros son las unidades de almacenamiento más pequeñas y rápidas dentro de la Unidad Central de Procesamiento (CPU). Contienen datos que la CPU está procesando activamente, como operandos para operaciones aritméticas, direcciones de memoria o punteros de instrucción. Su integración directa con la CPU permite el acceso inmediato, lo cual es crucial para ejecutar instrucciones a las velocidades más altas posibles sin una latencia significativa.

Memoria caché (L1, L2, L3)

La memoria caché actúa como un búfer de alta velocidad entre la CPU y la memoria principal (RAM). Está diseñada para almacenar datos e instrucciones a los que se accede con frecuencia, lo que reduce el tiempo que la CPU pasa esperando datos. La caché L1 es la más rápida y pequeña, directamente integrada en cada núcleo de la CPU. La caché L2 es más grande y ligeramente más lenta, a menudo dedicada por núcleo. La caché L3 es la más grande y lenta de los niveles de caché, normalmente compartida entre todos los núcleos de la CPU, lo que reduce aún más los tiempos de acceso a la memoria principal.

Memoria principal (RAM)

La memoria de acceso aleatorio (RAM) sirve como la memoria de trabajo principal de la computadora. Es mucho más grande que la caché, pero más lenta. La RAM contiene datos y programas que están actualmente en uso por la CPU, lo que permite una rápida recuperación y modificación. Cuando se inicia un programa, se carga desde el almacenamiento persistente (como un SSD o HDD) en la RAM para que la CPU acceda y ejecute. La RAM es volátil, lo que significa que su contenido se pierde cuando se apaga la alimentación.

Almacenamiento persistente (SSD y HDD)

Las unidades de estado sólido (SSD) y las unidades de disco duro (HDD) son formas de almacenamiento persistente no volátil, lo que significa que conservan los datos incluso cuando la computadora está apagada. Los SSD utilizan memoria flash y ofrecen velocidades de lectura/escritura significativamente más rápidas, tiempos de arranque más rápidos y una carga de aplicaciones más rápida en comparación con los HDD tradicionales. Los HDD, basados en platos magnéticos giratorios, son más lentos, pero proporcionan capacidades de almacenamiento mucho mayores a un costo menor por gigabyte, lo que los hace adecuados para el archivo de datos a largo plazo o el almacenamiento de archivos muy grandes.

A medida que avanzamos en la jerarquía, el almacenamiento se vuelve más lento, pero más grande y menos costoso. El objetivo es mantener los datos a los que se accede con frecuencia en tipos de memoria más rápidos.

Núcleos del sistema operativo

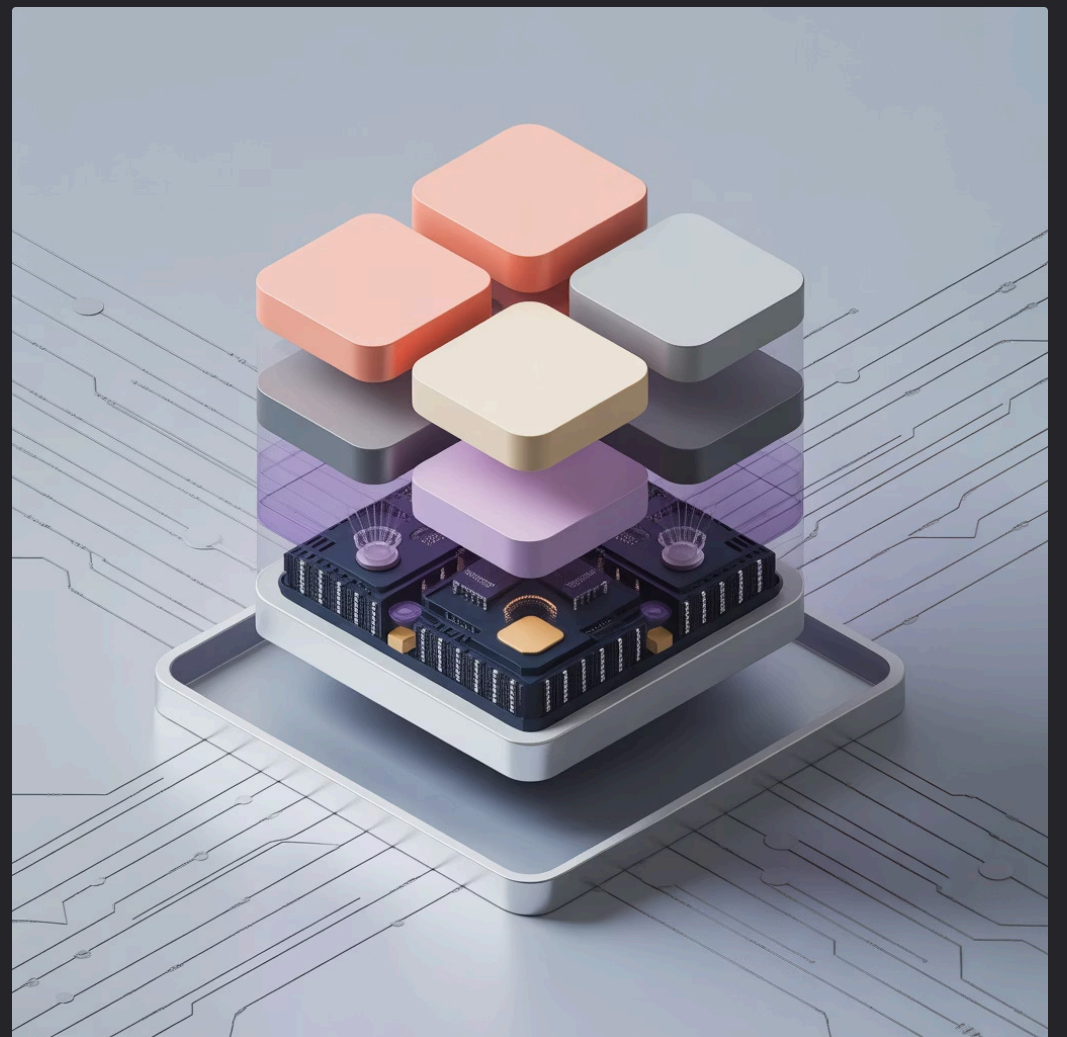
¿Qué es un núcleo?

El núcleo es el componente central de un sistema operativo, que actúa como el puente fundamental entre el hardware y el software. Tiene control total sobre todo en el sistema, gestionando los recursos esenciales y manejando las operaciones críticas del sistema desde el inicio hasta el apagado.

Es la primera parte del sistema operativo que se carga en la memoria y permanece residente durante todo el funcionamiento del ordenador, facilitando la comunicación entre el software de la aplicación y los componentes de procesamiento de datos del hardware.

Tipos de núcleo

- **Monolítico:** En este diseño, todos los servicios del sistema operativo (como la gestión de procesos, la gestión de la memoria, el sistema de archivos y los controladores de dispositivos) se ejecutan juntos en un único espacio de direcciones dentro del modo núcleo. Este enfoque ofrece un alto rendimiento debido a las llamadas directas a funciones, pero puede ser menos modular, de mayor tamaño y más difícil de mantener (por ejemplo, Linux, los sistemas Unix tradicionales).
- **Microkernel:** Esta arquitectura presenta un núcleo mínimo que proporciona sólo los servicios esenciales como la comunicación entre procesos (IPC) y la gestión básica de la memoria. Otros servicios, como los sistemas de archivos y los controladores de dispositivos, se ejecutan como procesos en el espacio de usuario. Esto mejora la modularidad, la fiabilidad y la seguridad, pero puede introducir una sobrecarga de rendimiento debido al aumento de la IPC entre los servicios (por ejemplo, MINIX, GNU Hurd).
- **Híbrido:** Como su nombre indica, se trata de una combinación de los enfoques monolítico y de microkernel, que pretende equilibrar las ventajas de rendimiento de los núcleos monolíticos con la modularidad y la estabilidad de los microkernels. Los servicios clave suelen permanecer en el espacio del núcleo para obtener rendimiento, mientras que los menos críticos pueden ejecutarse como procesos en el espacio del usuario (por ejemplo, Windows NT, macOS).



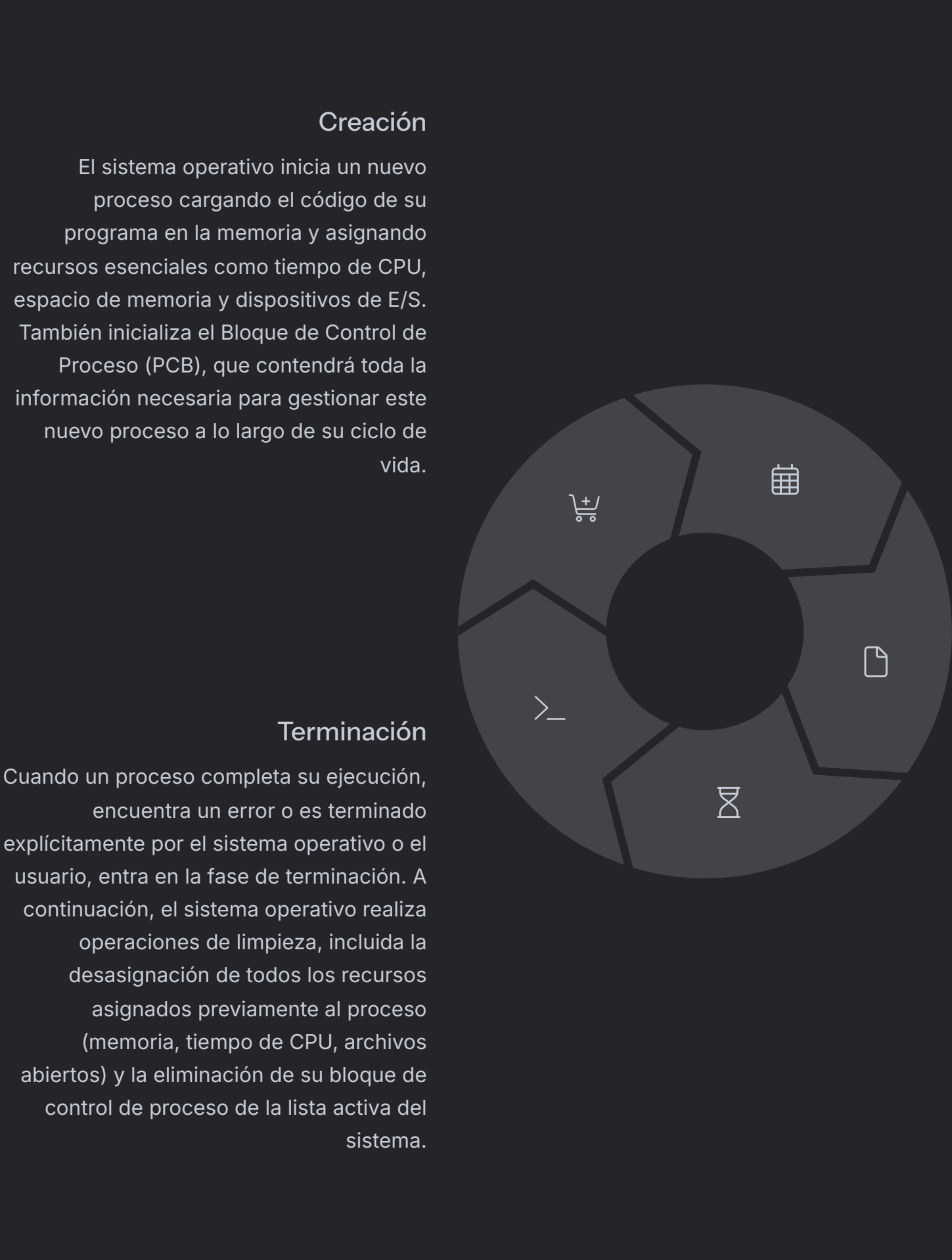
El núcleo opera en modo privilegiado, también conocido como modo núcleo, lo que le otorga acceso directo e ilimitado al hardware del sistema. Esto le permite realizar tareas cruciales que las aplicaciones de usuario no pueden. Proporciona varias funciones vitales, actuando como la máxima autoridad para los recursos del sistema:

- **Gestión de procesos:** Responsable de la creación, programación y finalización de los procesos, asignando eficientemente el tiempo de la CPU y los recursos para garantizar que varios programas puedan ejecutarse simultáneamente sin interferencias, y gestionando los estados de los procesos.
- **Gestión de la memoria:** Gestiona la memoria del ordenador, asignando y desasignando espacio de memoria para los programas en ejecución, gestionando la memoria virtual y garantizando la protección de la memoria entre los procesos para evitar conflictos y corrupción de datos.
- **Controladores de dispositivos:** Actúa como intermediario, traduciendo las peticiones del sistema operativo o de las aplicaciones en comandos que los dispositivos de hardware específicos (como impresoras, teclados o tarjetas de red) puedan entender, facilitando así las operaciones de entrada/salida (E/S).
- **Interfaz de llamadas al sistema:** Proporciona una interfaz de programación de aplicaciones (API) definida a través de la cual los programas de nivel de usuario pueden solicitar servicios del núcleo, como la E/S de archivos, la creación de procesos o la comunicación de red, de forma segura y controlada.

Comprender el papel del núcleo y sus diversos tipos es fundamental para comprender cómo funcionan los sistemas operativos e interactúan eficazmente con el hardware subyacente. Es realmente el corazón del sistema, orquestando todas las operaciones de bajo nivel y la asignación de recursos.

Gestión de procesos

La gestión de procesos es una función central del sistema operativo que se encarga de la creación, programación, ejecución y finalización de los procesos. Un proceso es esencialmente una instancia de un programa en ejecución, que abarca el código del programa, su actividad actual y sus recursos. El sistema operativo garantiza una asignación justa del tiempo de CPU y los recursos entre múltiples procesos concurrentes, manteniendo la estabilidad y la capacidad de respuesta del sistema.



Creación

El sistema operativo inicia un nuevo proceso cargando el código de su programa en la memoria y asignando recursos esenciales como tiempo de CPU, espacio de memoria y dispositivos de E/S. También inicializa el Bloque de Control de Proceso (PCB), que contendrá toda la información necesaria para gestionar este nuevo proceso a lo largo de su ciclo de vida.

Terminación

Cuando un proceso completa su ejecución, encuentra un error o es terminado explícitamente por el sistema operativo o el usuario, entra en la fase de terminación. A continuación, el sistema operativo realiza operaciones de limpieza, incluida la desasignación de todos los recursos asignados previamente al proceso (memoria, tiempo de CPU, archivos abiertos) y la eliminación de su bloque de control de proceso de la lista activa del sistema.

Programación

El sistema operativo determina qué proceso tiene acceso a la CPU y durante cuánto tiempo. Esto se gestiona mediante algoritmos de programación que tienen como objetivo optimizar el rendimiento, la equidad y el tiempo de respuesta del sistema. El programador selecciona un proceso de la cola de listos y lo envía para su ejecución en la CPU, gestionando los cambios de contexto entre procesos.

Ejecución

Durante la fase de ejecución, el proceso se ejecuta en la CPU, realizando sus tareas designadas mediante la ejecución de instrucciones. Puede acceder a su memoria asignada e interactuar con los dispositivos de E/S. El sistema operativo supervisa continuamente el estado del proceso, asegurándose de que funciona dentro de sus recursos asignados y se adhiere a las políticas del sistema.

Espera

Un proceso puede entrar en un estado de espera si requiere un recurso que no está disponible actualmente o si necesita completar una operación de E/S (por ejemplo, leer del disco o la entrada del usuario). Mientras espera, el proceso renuncia temporalmente a la CPU, lo que permite que otros procesos se ejecuten. Una vez que el recurso requerido está disponible o la operación de E/S se completa, el proceso vuelve al estado de listo.

El sistema operativo mantiene un bloque de control de proceso (PCB) completo para cada proceso. Esta estructura de datos vital contiene información crítica, incluyendo el estado del proceso (por ejemplo, en ejecución, en espera, listo), el contador de programa (dirección de la siguiente instrucción), los registros de la CPU, la información de gestión de la memoria, una lista de archivos abiertos, la información del estado de E/S y los datos de contabilidad. El PCB es esencial para permitir la multitarea y garantizar el aislamiento adecuado y la gestión de recursos para cada programa.

Memoria virtual

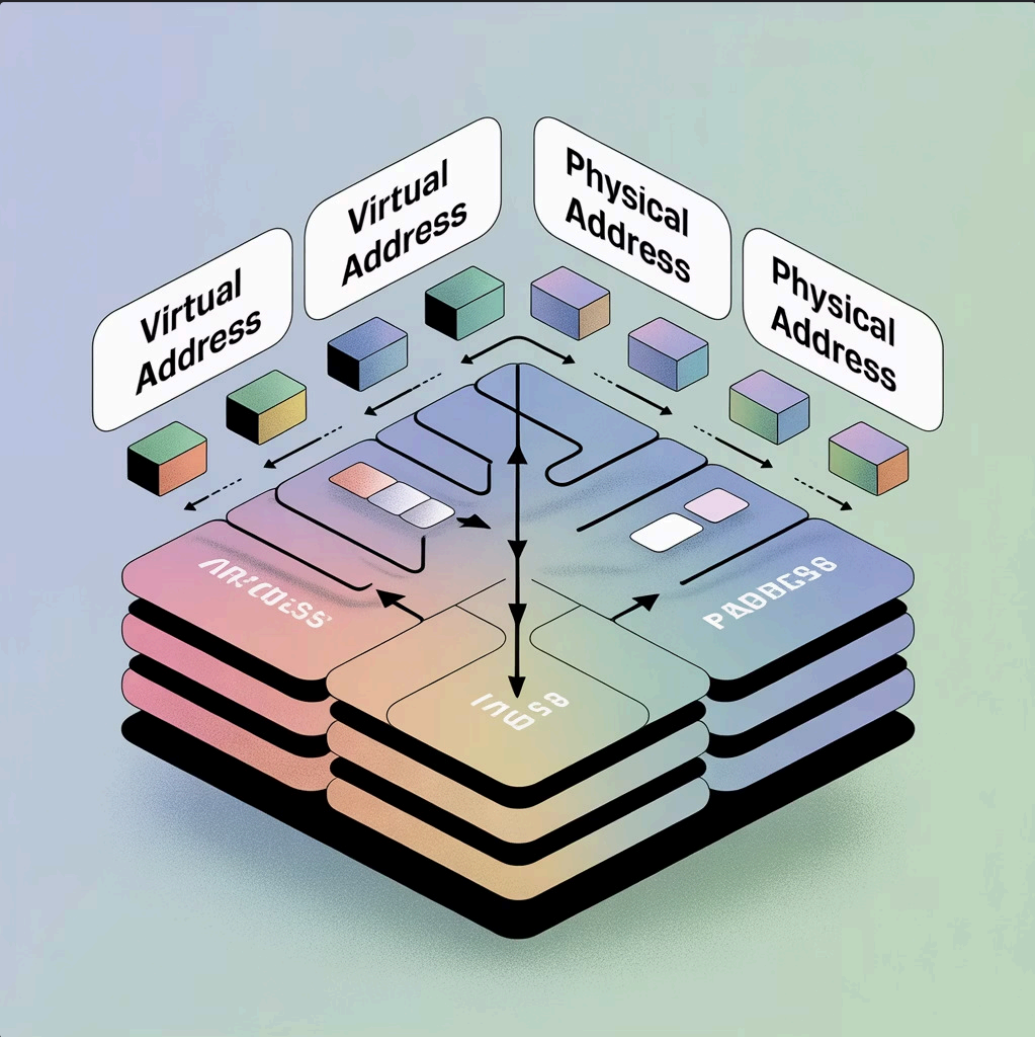
La memoria virtual es una técnica de administración de memoria que permite a una computadora compensar la escasez de memoria física transfiriendo temporalmente datos de la memoria de acceso aleatorio (RAM) al almacenamiento en disco. Crea la ilusión de que los procesos tienen un bloque de memoria grande y contiguo, aislándolos de las complejidades de la administración de la memoria física.

Cómo funciona la memoria virtual

La memoria virtual divide la RAM física en bloques de tamaño fijo llamados "páginas" y crea una asignación entre las direcciones virtuales (utilizadas por los programas) y las direcciones físicas (ubicaciones reales de la RAM). Esta asignación es administrada por el sistema operativo a través de una estructura de datos llamada **tabla de páginas**, que traduce las direcciones lógicas generadas por la CPU en direcciones físicas en la RAM. Cuando un programa solicita datos, la **Unidad de administración de memoria (MMU)** comprueba la tabla de páginas para encontrar la dirección física correspondiente. Si la página no está en la RAM, se produce un fallo de página.

Beneficios

- Los programas pueden usar más memoria de la disponible físicamente: esto permite ejecutar aplicaciones más grandes o más aplicaciones simultáneamente de lo que sería posible solo con la RAM física.
- Protección de memoria entre procesos: cada proceso tiene su propio espacio de direcciones virtuales, lo que evita que un programa acceda directamente o dañe la memoria de otro.
- Utilización eficiente de la memoria: el sistema operativo puede cargar solo las partes necesarias de un programa en la RAM, liberando memoria física para otros procesos y reduciendo la huella de memoria general. Las páginas también se pueden compartir entre procesos, lo que optimiza aún más el uso.



Fallos de página

Se produce un fallo de página cuando un programa intenta acceder a una página de memoria virtual que actualmente no está cargada en la RAM física del sistema. Cuando esto sucede:

1. La CPU activa un fallo de página: la MMU detecta que la página virtual requerida no está asignada a una página física.
2. El sistema operativo carga la página necesaria desde el disco: el sistema operativo intercepta el fallo de página, localiza la página requerida en el disco duro (en el espacio de intercambio) y la carga en un marco disponible en la memoria física.
3. La ejecución continúa: una vez que se carga la página, se actualiza la tabla de páginas y la CPU vuelve a ejecutar la instrucción que causó el fallo de página, lo que permite que el programa continúe como si la página siempre estuviera en la RAM. Los fallos de página frecuentes pueden degradar significativamente el rendimiento del sistema debido a la lenta E/S del disco.

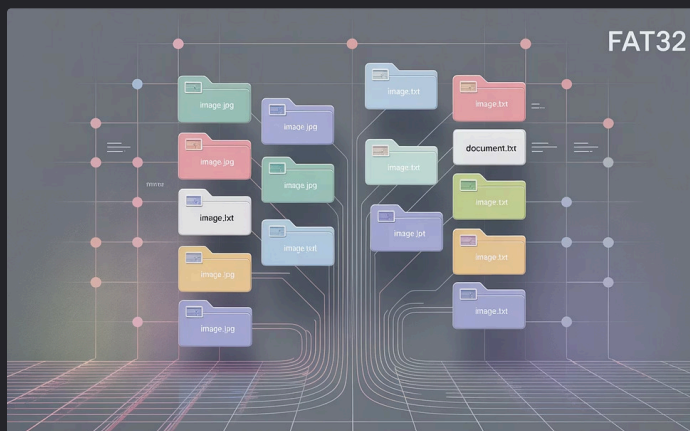
Archivo de intercambio y paginación

Para admitir la memoria virtual, los sistemas operativos utilizan una parte del disco duro llamada "espacio de intercambio" o "archivo de paginación". Cuando la RAM física está llena, el sistema operativo mueve (intercambia) las páginas que se usan con menos frecuencia de la RAM al espacio de intercambio en el disco. Por el contrario, cuando un programa necesita una página que está en el espacio de intercambio, se "intercambia" a la RAM, a menudo reemplazando otra página que luego se intercambia. Este movimiento constante de páginas entre la RAM y el disco es fundamental para el funcionamiento de la memoria virtual.

La memoria virtual es un componente crucial de los sistemas operativos modernos, ya que permite una multitarea sólida, la protección de la memoria y la capacidad de ejecutar aplicaciones que exigen más memoria de la instalada físicamente. Si bien mejora la flexibilidad del sistema, la dependencia excesiva del intercambio puede provocar cuellos de botella en el rendimiento, una condición conocida como "thrashing".

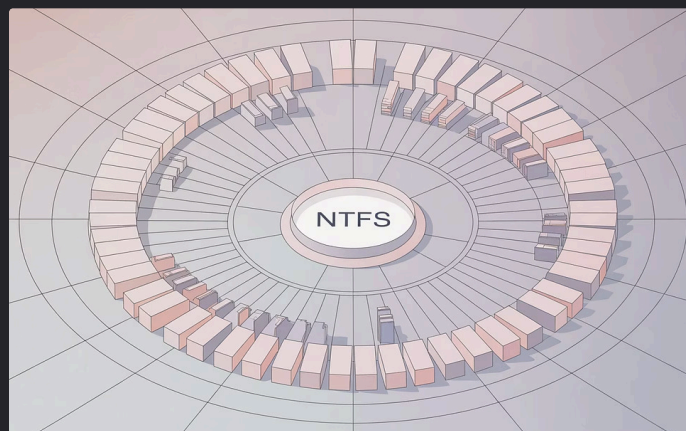
Sistemas de archivos

Los sistemas de archivos son componentes fundamentales de un sistema operativo, responsables de cómo se almacenan, recuperan y organizan los datos en varios dispositivos de almacenamiento, como discos duros, SSD y unidades flash USB. Proporcionan la estructura lógica que permite a los usuarios y las aplicaciones interactuar con los datos de una manera intuitiva, abstrayendo la compleja disposición física de los bits en el medio de almacenamiento. Sin un sistema de archivos, los datos serían simplemente un gran bloque de información no administrada.



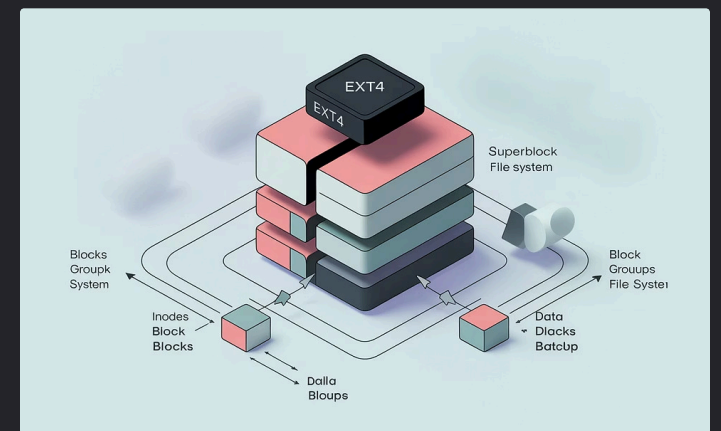
FAT32

FAT32 (Tabla de asignación de archivos 32) es un sistema de archivos heredado conocido por su amplia compatibilidad con diferentes sistemas operativos y dispositivos, incluidas computadoras antiguas, consolas de juegos y la mayoría de las unidades USB. A pesar de su amplio alcance, tiene limitaciones significativas: los archivos individuales no pueden exceder los 4 GB y las particiones generalmente están limitadas a 32 GB en Windows, lo que lo hace inadecuado para archivos modernos grandes como videos HD o instalaciones de software grandes. También carece de características avanzadas como el registro por diario o los permisos.



NTFS

NTFS (Sistema de archivos de nueva tecnología) es el sistema de archivos principal y más avanzado utilizado por los sistemas operativos Microsoft Windows. Ofrece mejoras significativas con respecto a FAT32, incluido el soporte para archivos y particiones de disco extremadamente grandes. Las características clave de NTFS incluyen permisos de archivos y carpetas (Listas de control de acceso) para mayor seguridad, cuotas de disco para administrar el almacenamiento del usuario, compresión de archivos, cifrado (EFS) y registro por diario robusto. El registro por diario garantiza la integridad de los datos y una recuperación más rápida de las fallas del sistema al registrar los cambios antes de que se escriban en el disco principal.



ext4

ext4 (Cuarto sistema de archivos extendido) es el sistema de archivos de registro por diario predeterminado y más utilizado para las distribuciones de Linux. Se basa en sus predecesores (ext2 y ext3) con mejoras notables en rendimiento, escalabilidad y confiabilidad. ext4 admite tamaños de archivo muy grandes (hasta 16 TB) y volúmenes (hasta 1 EB), lo que lo hace adecuado para las necesidades modernas de almacenamiento de alta capacidad. Características como la asignación basada en extensiones, la asignación retrasada y las comprobaciones más rápidas del sistema de archivos contribuyen a su eficiencia, mientras que el registro por diario proporciona coherencia de los datos y una recuperación rápida después de una pérdida de energía inesperada o fallas del sistema.

Cada uno de estos sistemas de archivos está optimizado para diferentes entornos y propósitos informáticos. Comprender sus características, como la compatibilidad, el conjunto de características y el rendimiento, es esencial para una gestión eficaz de los datos, la seguridad del sistema y la eficiencia informática general. La elección del sistema de archivos a menudo depende del sistema operativo en uso y de los requisitos específicos para el almacenamiento y el acceso a los datos.

Proceso de arranque



Encendido

Cuando se enciende el ordenador, la Unidad Central de Procesamiento (CPU) comienza inmediatamente a ejecutar código desde una ubicación de memoria especial no volátil. Esta memoria contiene el firmware del Sistema Básico de Entrada/Salida (BIOS) o la Interfaz de Firmware Extensible Unificada (UEFI), que es el primer software que se ejecuta, iniciando la secuencia de inicio del sistema.



POST

El BIOS/UEFI ejecuta la autoprueba de encendido (POST) para realizar una comprobación preliminar de los componentes de hardware esenciales. Esto incluye la verificación de la CPU, la memoria (RAM), la tarjeta gráfica, el teclado y otros dispositivos críticos. Si se detecta algún problema importante durante la POST, el sistema normalmente se detiene y señala un error, a menudo a través de una serie de pitidos o mensajes en pantalla.



Cargador de arranque

Después de una POST exitosa, el BIOS/UEFI localiza y carga un pequeño programa llamado cargador de arranque. Este programa se encuentra normalmente en el dispositivo de arranque (por ejemplo, disco duro, SSD o unidad USB) en un área específica como el Registro de Arranque Maestro (MBR) o la Tabla de Particiones GUID (GPT). La función principal del cargador de arranque es encontrar y cargar el kernel del sistema operativo, mucho más grande, en la memoria, pasando el control a este.



Inicialización del kernel

Una vez cargado, el kernel del sistema operativo toma el control. Durante esta fase, inicializa los servicios centrales del sistema, configura la gestión de la memoria, configura la programación de procesos y carga los controladores de dispositivos esenciales necesarios para los componentes de hardware cruciales. Esto hace que las partes fundamentales del sistema operativo estén listas para ejecutarse y gestionar los recursos del ordenador.



Espacio de usuario

La etapa final del proceso de arranque implica la transición del entorno de solo kernel al espacio de usuario. Aquí, el kernel lanza los procesos iniciales a nivel de usuario, como el demonio de inicialización del sistema (por ejemplo, systemd en Linux o el Administrador de Sesiones de Windows). Estos procesos inician entonces otros servicios de usuario, el administrador de inicio de sesión y, en última instancia, la interfaz gráfica de usuario, haciendo que el sistema esté listo para la interacción del usuario y la ejecución de aplicaciones.

Multitarea y Programación

Tipos de Multitarea

- **Cooperativa:** En la multitarea cooperativa, se espera que los procesos cedan voluntariamente el control de la CPU a otros procesos. Este enfoque se basa en que cada programa se comporte correctamente y renuncie regularmente al control, lo que puede provocar la congelación del sistema si un solo proceso funciona mal o se atasca en un bucle infinito.
- **Preferente:** Este es el enfoque moderno utilizado por la mayoría de los sistemas operativos. El sistema operativo toma el control de la asignación de la CPU y puede interrumpir los procesos en cualquier momento (por ejemplo, mediante temporizadores o interrupciones de hardware). Esto garantiza la equidad, evita que una sola aplicación monopolice la CPU y mejora la capacidad de respuesta general del sistema.

Algoritmos de Programación

- **Primero en Llegar, Primero en Ser Servido (FCFS):** Este es el algoritmo de programación más simple, donde los procesos se ejecutan en el orden en que llegan. Si bien es sencillo, FCFS puede ser ineficiente si un proceso de larga duración llega primero, lo que lleva a un "efecto de convoy" donde muchos procesos más cortos esperan innecesariamente.
- **Round Robin:** En la programación Round Robin, a cada proceso se le asigna una porción de tiempo fija (quantum) durante la cual se le permite ejecutarse. Si el proceso no se completa dentro de su porción de tiempo, se interrumpe y se mueve al final de la cola de espera. Este método proporciona una asignación justa del tiempo de CPU entre todos los procesos activos.
- **Basado en la Prioridad:** A los procesos se les asigna un nivel de prioridad, y el programador siempre selecciona el proceso con la prioridad más alta para ejecutarlo. Si bien es eficaz para las tareas críticas, los procesos de menor prioridad pueden sufrir de "inanición" si un flujo continuo de procesos de mayor prioridad les impide obtener tiempo de CPU.



Los sistemas operativos modernos utilizan algoritmos complejos que se ajustan dinámicamente en función de varios factores para optimizar el rendimiento, la equidad y la capacidad de respuesta. Estas consideraciones incluyen:

- **Prioridad del proceso:** Las prioridades pueden ser estáticas o dinámicas, a menudo ajustadas por el sistema operativo para evitar la inanición (por ejemplo, aumentando la prioridad de los procesos que han estado esperando durante mucho tiempo).
- **Historial de uso de la CPU:** El programador puede rastrear cuánto tiempo de CPU ha consumido recientemente un proceso. A los procesos que han usado menos CPU se les podría dar mayor prioridad o porciones de tiempo más largas para garantizar una distribución equilibrada de los recursos.
- **Procesos limitados por E/S frente a procesos limitados por CPU:** Los procesos limitados por E/S pasan más tiempo esperando las operaciones de E/S (como leer de un disco), mientras que los procesos limitados por CPU pasan la mayor parte de su tiempo realizando cálculos. Los programadores a menudo priorizan los procesos limitados por E/S para mantener los dispositivos de E/S ocupados y mejorar el rendimiento general del sistema.
- **Equidad y capacidad de respuesta:** Un buen programador tiene como objetivo proporcionar una parte justa del tiempo de CPU a todos los procesos, al tiempo que garantiza que las aplicaciones interactivas sigan respondiendo a la entrada del usuario. Esto a menudo implica compensaciones entre maximizar la utilización de la CPU y minimizar la latencia para ciertas tareas.

La elección del algoritmo de programación y sus parámetros impacta significativamente el rendimiento percibido y la eficiencia de un sistema operativo, especialmente en entornos multiusuario o altamente concurrentes.

Consideraciones de seguridad



Aislamiento del usuario

Los procesos se ejecutan con permisos de usuario específicos, lo que evita el acceso no autorizado a los recursos del sistema o a los datos de otros usuarios.

Este mecanismo es crucial para los sistemas multiusuario, ya que garantiza que un fallo o una actividad maliciosa en la aplicación de un usuario no comprometa todo el sistema ni la privacidad de otro usuario.



Protección de la memoria

El sistema operativo evita que los procesos accedan a la memoria asignada a otros procesos, evitando así bloqueos y brechas de seguridad.

Técnicas como la memoria virtual y la aleatorización del diseño del espacio de direcciones (ASLR) dificultan a los atacantes la predicción de las ubicaciones de la memoria y la explotación de vulnerabilidades para la escalada de privilegios o el robo de datos.



Niveles de privilegio

Las CPU modernas admiten diferentes anillos de privilegio (modo kernel frente a modo usuario) para restringir el acceso a operaciones confidenciales.

El modo kernel tiene acceso completo al hardware y a los recursos del sistema, mientras que el modo usuario restringe las operaciones. Las llamadas al sistema proporcionan una interfaz controlada para que las aplicaciones en modo usuario soliciten servicios a nivel del kernel, manteniendo así la integridad del sistema.



Arranque seguro

Verifica criptográficamente cada componente del proceso de arranque para garantizar que el sistema no haya sido manipulado.

A menudo, basándose en el firmware UEFI y los módulos de plataforma segura (TPM), el arranque seguro garantiza que solo se pueda cargar software de confianza y firmado durante el inicio, protegiendo contra rootkits y malware de tiempo de arranque.



Control de acceso

Los sistemas operativos aplican controles de acceso granulares (por ejemplo, listas de control de acceso, permisos de archivo) en archivos, directorios y otros recursos del sistema.

Esto garantiza que solo los usuarios o procesos autorizados puedan realizar acciones específicas como leer, escribir o ejecutar datos, lo que reduce significativamente el riesgo de modificación o divulgación no autorizada de datos.



Registro de auditoría

El sistema operativo registra los eventos críticos del sistema, las acciones relacionadas con la seguridad y las actividades del usuario en registros detallados.

Estos registros de auditoría exhaustivos son invaluable para detectar comportamientos sospechosos, realizar análisis forenses después de un incidente de seguridad y demostrar el cumplimiento de las políticas y regulaciones de seguridad.