Introducción a los Sistemas Operativos

UNIDAD 2: Introducción a los sistemas operativos

- 2.1. Definición de un sistema operativo.
- 2.2. Componentes del sistema.
- 2.3. Servicios del sistema operativo.
- 2.4. Llamadas al sistema.
- 2.5. Programas del sistema.

2.1. Definición de Sistema Operativo

CONCEPTO CENTRAL

El SO como intermediario y gestor.

Definición

Programa que administra el hardware de una computadora y actúa como interfaz entre el usuario y los componentes físicos.

Gestor de recursos

Asignación de CPU, memoria, E/S y almacenamiento.

Programa de control

Gestiona la ejecución de programas de usuario para evitar errores y uso incorrecto del sistema.

Abstracción

Proporciona una "máquina virtual" más sencilla de programar que el hardware crudo.

2.2. Componentes del Sistema

El sistema operativo gestiona cinco grandes áreas de responsabilidad. Cada componente actúa como una pieza del rompecabezas que, en conjunto, hace posible que el hardware y el software coexistan de forma ordenada.

Gestión de Procesos

Creación, planificación (scheduling) y sincronización de procesos. Decide qué proceso ocupa la CPU en cada momento.

Gestión de Memoria

Asignación y liberación de espacio en memoria principal. Controla qué proceso accede a qué zona de RAM.

Gestión de Archivos

Organización de directorios y manipulación de datos en disco. Ofrece una vista lógica del almacenamiento.

Gestión de E/S

Sistema de buffering, caché y drivers. Abstrae la complejidad de los dispositivos periféricos.

Almacenamiento Secundario

Optimización del espacio en disco. Gestiona la memoria libre y la ubicación física de los datos.

2.3. Servicios del Sistema Operativo

Los servicios del SO son las funciones que ofrece directamente a usuarios y programas. Son la razón por la que un SO existe: sin ellos, cada programa tendría que gestionar el hardware por su cuenta, lo que sería inseguro e inviable.

Interfaz de Usuario

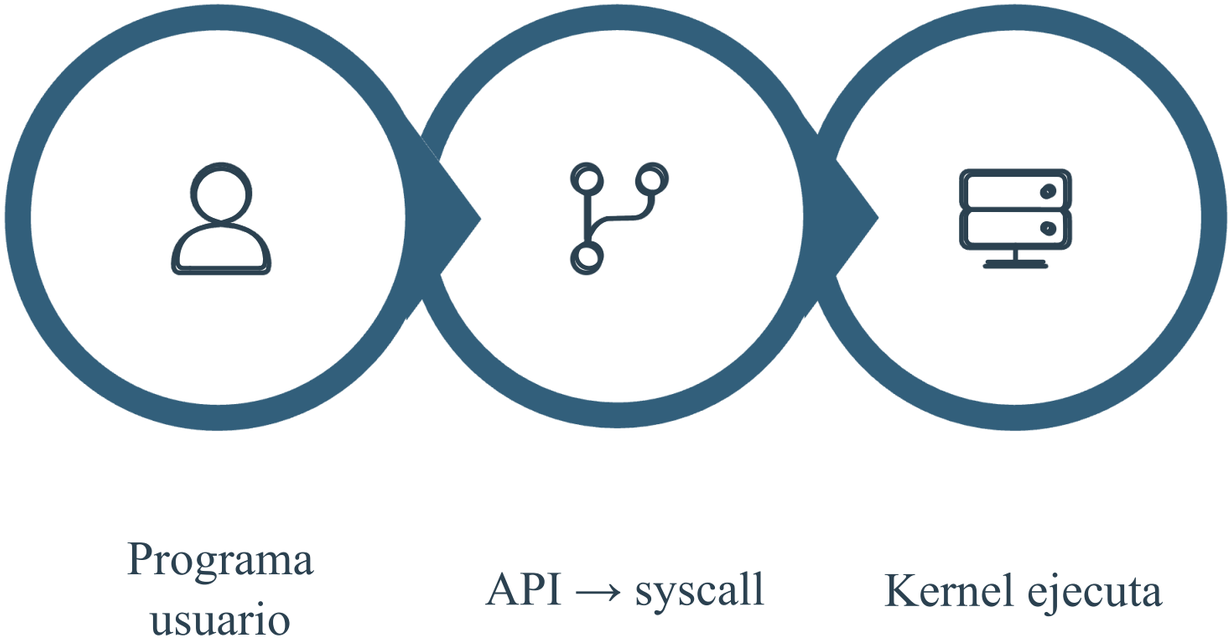
*El SO proporciona tres tipos principales de interfaz: **CLI** (línea de comandos, como bash o PowerShell), **GUI** (gráfica, como Windows o GNOME) e interfaces **táctiles** (móviles y tablets). La CLI es preferida en servidores por su eficiencia.*

Resto de Servicios Clave

- ***Ejecución de programas:*** *Carga el programa en memoria y cede el control a la CPU.*
- ***Operaciones de E/S:*** *Los programas no acceden directamente al hardware; el SO actúa como intermediario.*
- ***Comunicaciones (IPC):*** *Intercambio de información entre procesos, ya sea en la misma máquina o en red.*
- ***Detección de errores:*** *Reacciona ante fallos de hardware (disco dañado) o software (división por cero, falta de papel).*

2.4. Llamadas al Sistema (System Calls)

*Las **llamadas al sistema** son el punto de entrada oficial a los servicios del SO. Un programa en modo usuario no puede acceder directamente al hardware; debe solicitar al kernel que lo haga en su nombre mediante una llamada al sistema.*



El diagrama muestra el camino completo desde que un programa solicita un servicio hasta que el kernel lo ejecuta y devuelve el resultado al proceso solicitante.

API: la capa intermedia

*Los programadores raramente invocan llamadas al sistema directamente. Usan **APIs** como **Win32** (Windows) o **POSIX** (Unix/Linux), que encapsulan las llamadas reales y ofrecen una interfaz más amigable y portable.*

Paso de Parámetros al Kernel

*Existen tres métodos para pasar datos al kernel: a través de **registros** de la CPU, mediante una **tabla en memoria** compartida, o apilando los parámetros en la **pila (stack)** del proceso.*

2.5. Programas del Sistema

*Los programas del sistema son herramientas que se distribuyen con el SO pero **no forman parte del kernel**. Son utilidades que facilitan la gestión del sistema, el desarrollo de software y la monitorización del estado del equipo.*

Manipulación de Archivos

Comandos como `copy`, `move` y `delete` permiten gestionar el sistema de ficheros sin necesidad de escribir código.

Información de Estado

Utilidades que muestran la fecha, hora, memoria libre, procesos activos y usuarios conectados. Ejemplos: `top`, `ps`, `df`.

Soporte a Lenguajes

Compiladores, depuradores e intérpretes forman parte del ecosistema del SO, aunque no son el SO en sí mismos.

Carga y Enlace

*Los **cargadores (loaders)** sitúan el programa en memoria para su ejecución; los **editores de enlace (linkers)** resuelven referencias entre módulos.*

Estructura de Sistema Operativo

Un sistema operativo no es un bloque monolítico de código: es una arquitectura cuidadosamente diseñada compuesta por componentes, servicios y estructuras que trabajan en conjunto para gestionar el hardware y ofrecer un entorno de ejecución seguro y eficiente. En esta unidad exploraremos cómo está organizado internamente un SO.

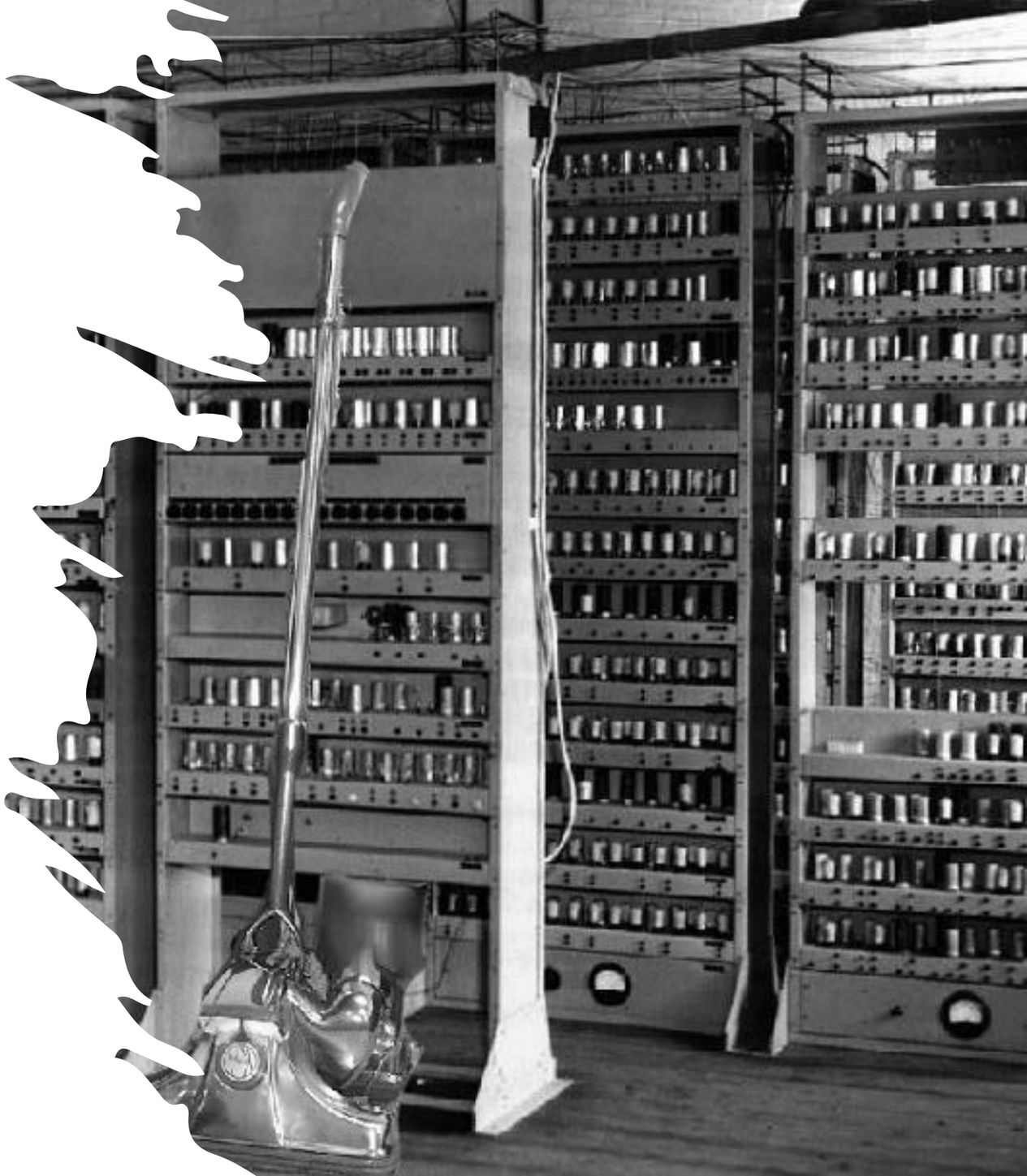
UNIDAD 3. Estructuras del sistema operativo:

1. Los primeros sistemas.
2. Sistemas por lotes (batch) sencillos.
3. Sistemas por lotes bajo multiprogramación.
4. Sistemas de tiempo compartido.
5. Sistemas de tiempo real.
6. Sistemas de computación personal.
7. Sistemas Cliente Servidor.
8. Estructura del sistema.
9. Máquinas virtuales.
10. Diseño e implementación de sistemas.
11. Generación de sistemas.

3.1 Los Primeros Sistemas (Años 40 - 50)

- **Características:**
  - No existía el concepto de "Sistema Operativo".
  - Acceso solitario: Un solo usuario reservaba bloques de tiempo.
  - Programación en lenguaje de máquina absoluto.

- **Limitación:**
  - Enorme desperdicio de tiempo de CPU mientras el programador montaba cintas o configuraba el equipo.



3.2 Sistemas por Lotes (Batch) Sencillos

Maximizar el uso del procesador agrupando trabajos similares.

01

Funcionamiento

Los usuarios entregaban sus trabajos (tarjetas perforadas) a un operador. El operador agrupaba trabajos similares en "lotes" y los cargaba en una cinta.

02

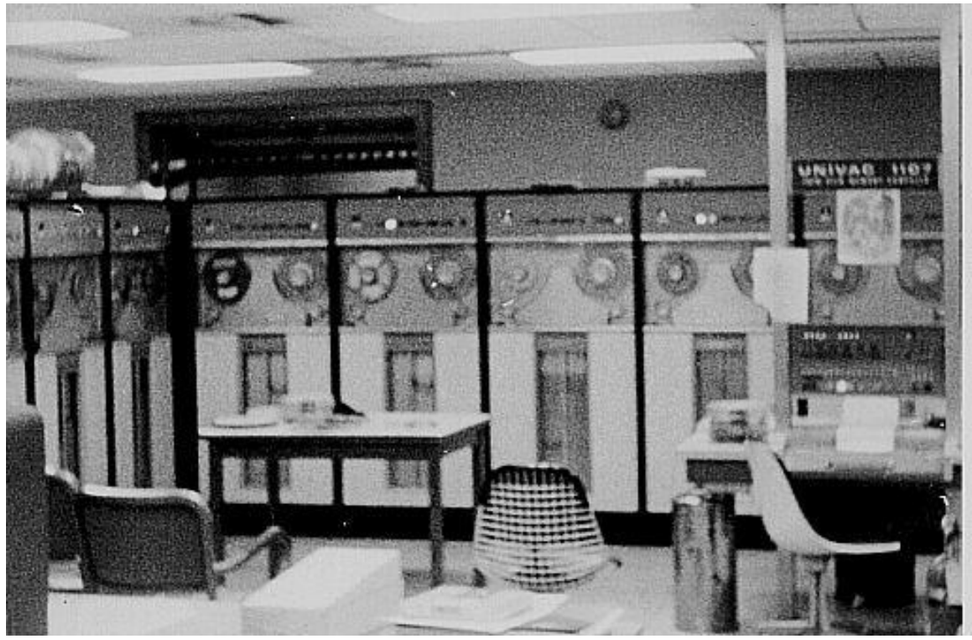
El Monitor Residente

El precursor del SO. Un programa que residía siempre en memoria y cargaba automáticamente el siguiente trabajo del lote.

03

Hito

*Aparece el concepto de **JCL (Job Control Language)**.*



4. Sistemas por Lotes bajo Multiprogramación

Mantener la CPU ocupada siempre que haya un proceso listo.

Mecanismo

La memoria alberga varios trabajos a la vez. Cuando un trabajo debe esperar por una operación de E/S, el SO conmuta a otro trabajo.

Requisitos Técnicos

- *Gestión de memoria (particionamiento).*
- *Planificación de CPU (decidir qué proceso sigue).*
- *Protección de memoria (evitar que un proceso acceda a otro).*

3.3 Sistemas de Tiempo Compartido (Time-Sharing)

Extensión lógica de la multiprogramación para interactividad.

Funcionamiento

*La CPU se reparte entre múltiples usuarios mediante conmutaciones de contexto muy rápidas (**Quantum de tiempo**).*

Experiencia de Usuario

Cada usuario tiene la ilusión de poseer la máquina en exclusiva.

Hito

Introducción de sistemas de archivos en línea y comunicación de terminales.

3.4. Sistemas de Tiempo Real (Real-Time Systems)

El tiempo es un parámetro de corrección.

Definición: *Sistemas donde la respuesta debe ocurrir dentro de un límite temporal estricto.*

Hard Real-Time

El incumplimiento de un plazo es un fallo crítico del sistema (ej. frenos ABS, control de vuelo).

Soft Real-Time

Se priorizan tareas críticas, pero un retraso ocasional no es catastrófico (ej. streaming de video).

3.5 Sistemas de Computación Personal (PC)

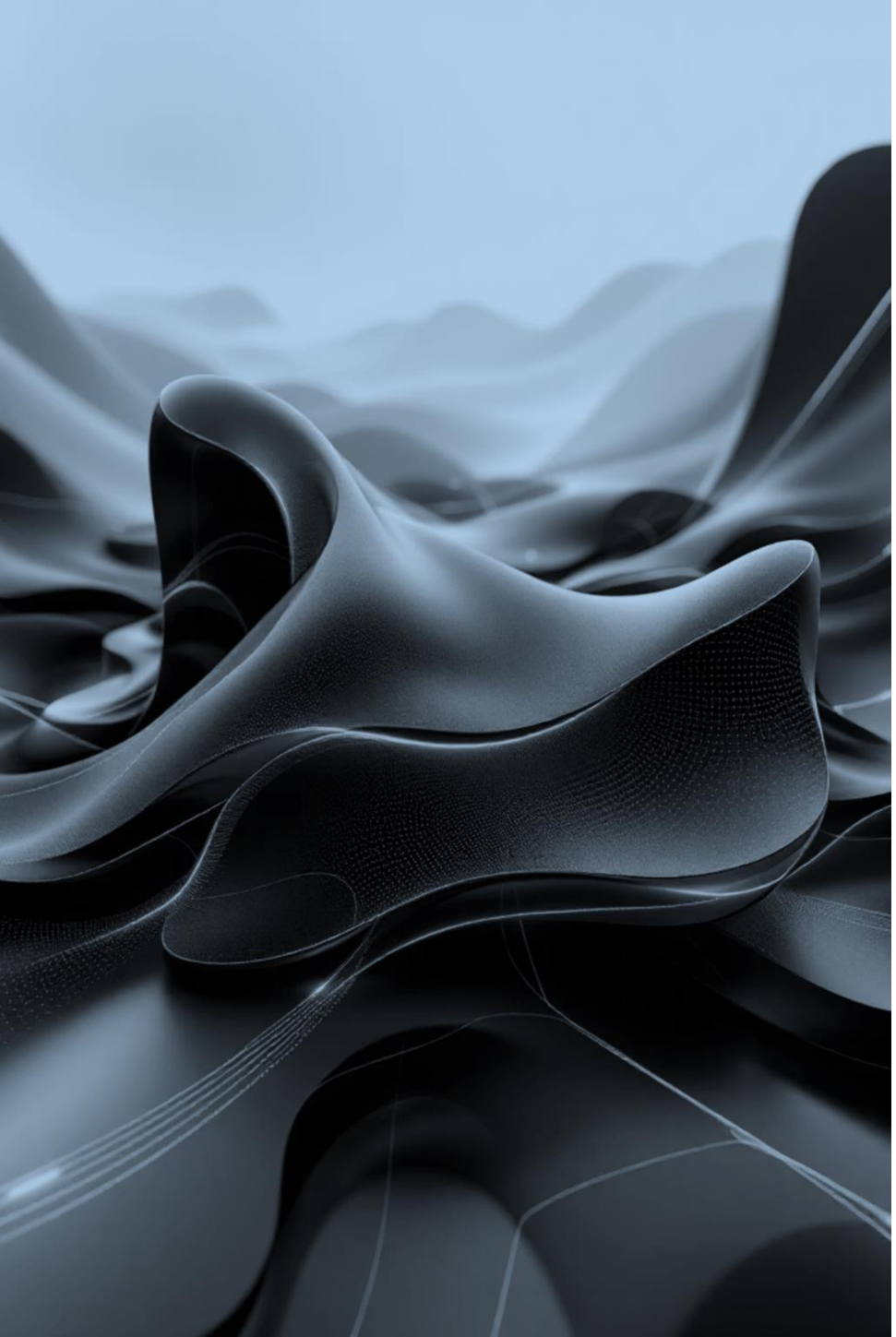
Optimizar la experiencia del usuario individual.

Evolución

*Surgimiento de microprocesadores. El foco se desplazó de la utilización máxima de la CPU a la **conveniencia y facilidad de uso**.*

Características

Interfaces Gráficas de Usuario (GUI), multitarea orientada al usuario y soporte para periféricos diversos.



3.6 Sistemas Cliente-Servidor

Distribución de tareas en red.

Arquitectura

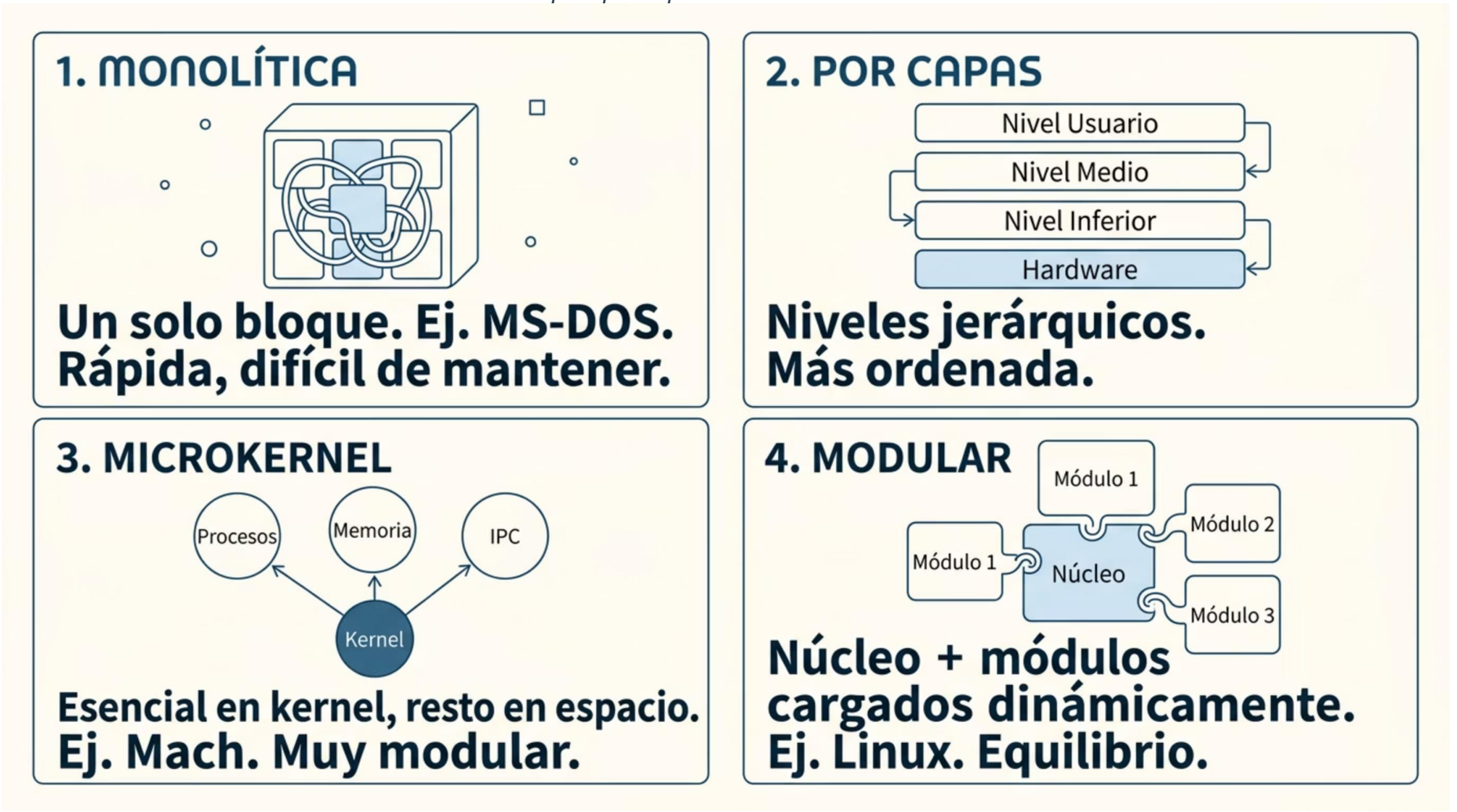
- ***Servidor:*** *Provee recursos o servicios (archivos, bases de datos, impresión).*
- ***Cliente:*** *Solicita servicios al servidor.*

Sistemas Operativos de Red

Capacidad para gestionar protocolos de comunicación, seguridad multiusuario y acceso remoto a recursos.

3.8 Estructura del Sistema

¿Cómo organizamos el código del SO? La elección de la estructura interna tiene implicaciones directas en el rendimiento, la seguridad y la mantenibilidad del sistema. Existen cuatro enfoques principales.



*Cada enfoque representa un compromiso diferente entre rendimiento, modularidad y complejidad. Los sistemas modernos como **Linux** y **Windows** adoptan una estructura **modular** que combina lo mejor del enfoque monolítico y del microkernel.*

2.7. Máquinas Virtuales (VM)

*Una **máquina virtual** es una abstracción creada por el SO que simula una computadora completa sobre el hardware físico. El componente clave es el **VMM (Virtual Machine Monitor)** o **hypervisor**, que crea una réplica exacta del hardware subyacente para cada máquina virtual.*

Tipos de Virtualización

Virtualización Completa

El SO invitado no sabe que está virtualizado. El hypervisor emula todo el hardware.

Paravirtualización

El SO invitado es consciente de la virtualización y colabora con el hypervisor para mejorar el rendimiento.

Contenedores (Docker)

Comparten el kernel del host pero aíslan el espacio de usuario. Más ligeros que las VM tradicionales.

Beneficios Clave

- ***Aislamiento total:** un fallo en una VM no afecta a las demás.*
- ***Seguridad:** entornos de prueba aislados del sistema principal.*
- ***Consolidación:** múltiples servidores en una sola máquina física.*
- ***Portabilidad:** las VM se pueden mover entre hosts fácilmente.*
- ***Snapshot:** capturas instantáneas del estado del sistema.*

2.8 Diseño e Implementación de Sistemas

Crear un sistema operativo implica tomar decisiones de diseño que equilibren las necesidades de los usuarios con las restricciones técnicas del sistema. Dos conceptos fundamentales guían estas decisiones.

Objetivos de Usuario

- *Fácil de usar y aprender*
- *Seguro y estable*
- *Rápido y responsivo*
- *Confiable ante fallos*

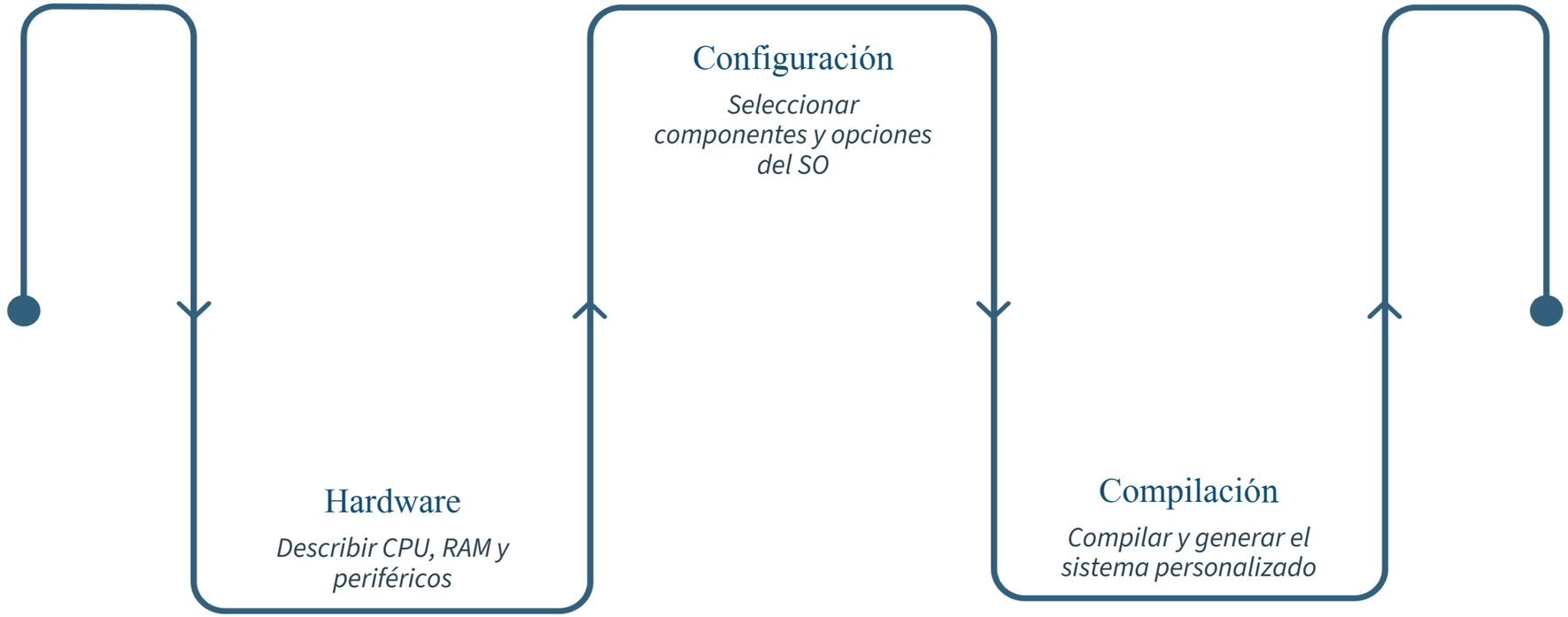
Objetivos del Sistema

- *Fácil de diseñar e implementar*
- *Confiable internamente*
- *Eficiente en uso de recursos*
- *Fácil de mantener y actualizar*

Principio clave — Mecanismos vs. Políticas: *El* ***mecanismo*** *define cómo hacer algo (ej. "el temporizador interrumpe cada 10 ms"). La* ***política*** *define qué se va a hacer (ej. "cada proceso tendrá 20 ms de CPU").* ***Separarlos es esencial*** *para lograr flexibilidad: el mismo mecanismo puede soportar políticas diferentes sin reescribir el código.*

2.8 Generación de Sistemas (SYSGEN)

*Un sistema operativo está diseñado para funcionar en una **clase de máquinas**, pero cada computadora concreta tiene características únicas: distinta CPU, cantidad de memoria, periféricos y configuración de disco. El proceso de adaptar el SO a un hardware específico se denomina **SYSGEN** (System Generation).*



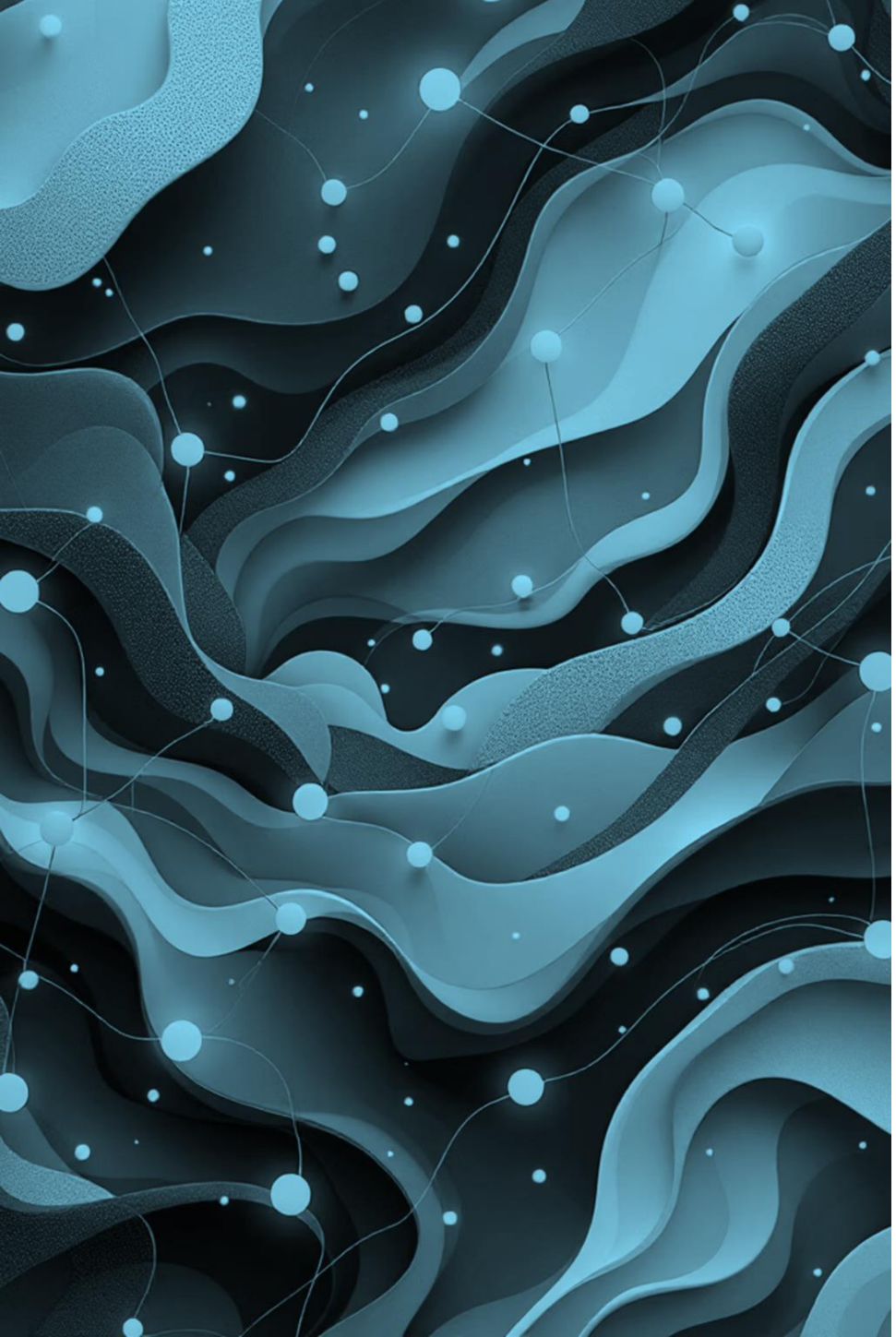
El proceso SYSGEN permite que un mismo SO base se adapte a entornos muy diversos, desde servidores de gran escala hasta dispositivos embebidos con recursos limitados.

¿Qué se configura en SYSGEN?

- *Tipo y número de CPUs*
- *Capacidad de memoria principal*
- *Dispositivos de E/S disponibles*
- *Opciones de red y almacenamiento*
- *Parámetros de planificación y seguridad*

Enfoques modernos

*En sistemas contemporáneos como **Linux**, el SYSGEN se realiza en tiempo de compilación del kernel o mediante la carga dinámica de módulos. En entornos embebidos, herramientas como **Yocto** o **Buildroot** automatizan este proceso para generar imágenes del sistema a medida.*



Resumen

Repasamos los conceptos clave que estructuran internamente un sistema operativo.

01

Definición del SO

Qué es un SO.

02

Componentes del SO

Procesos, memoria, archivos, E/S y almacenamiento secundario son las cinco áreas de gestión del kernel.

03

Servicios y System Calls

El SO ofrece servicios a través de interfaces (CLI, GUI) y las llamadas al sistema son el puente entre programas y kernel.

04

Estructuras del SO

Monolítica, por capas, microkernel y modular: cada arquitectura implica compromisos entre rendimiento y flexibilidad.

05

Virtualización y SYSGEN

Las VM crean entornos aislados sobre hardware compartido; SYSGEN adapta el SO al hardware concreto de cada máquina.

Resumen Comparativo de Sistemas Operativos

ANÁLISIS

Cada generación de sistemas operativos trajo mejoras significativas en el aprovechamiento de recursos y la experiencia del usuario. Aquí un resumen:

Sistema	¿Quién manda?	Aprovechamiento CPU	Tiempo de Respuesta
Primeros Sistemas	El programador	Muy Bajo (manual)	Días (espera de turno)
Lotes (Batch)	El Operador	Medio	Horas
Multiprogramación	El SO (Planificador)	Alto	Minutos
Tiempo Compartido	El Usuario (Simultáneo)	Muy Alto	Milisegundos