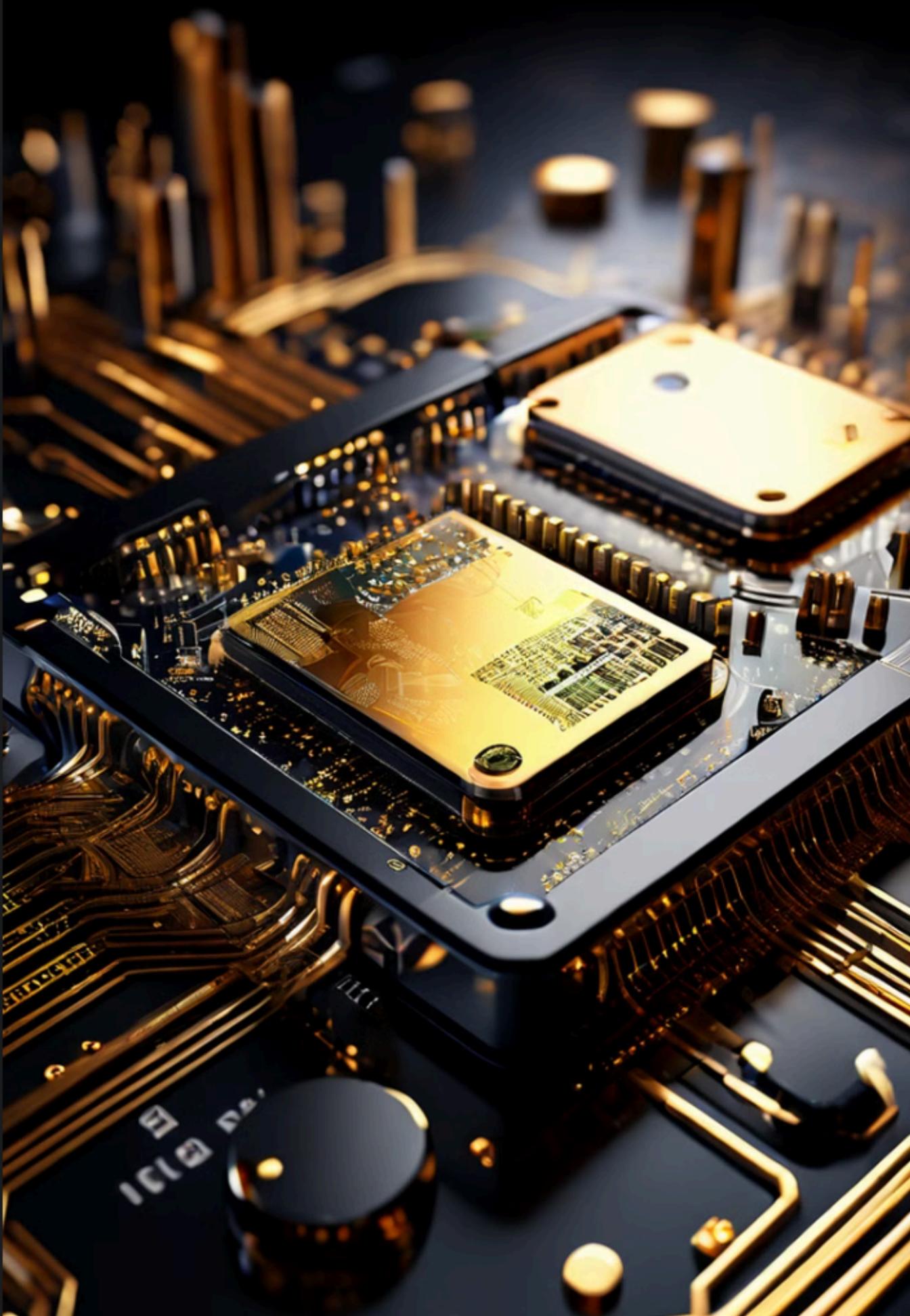


# Descripción y Control de Procesos



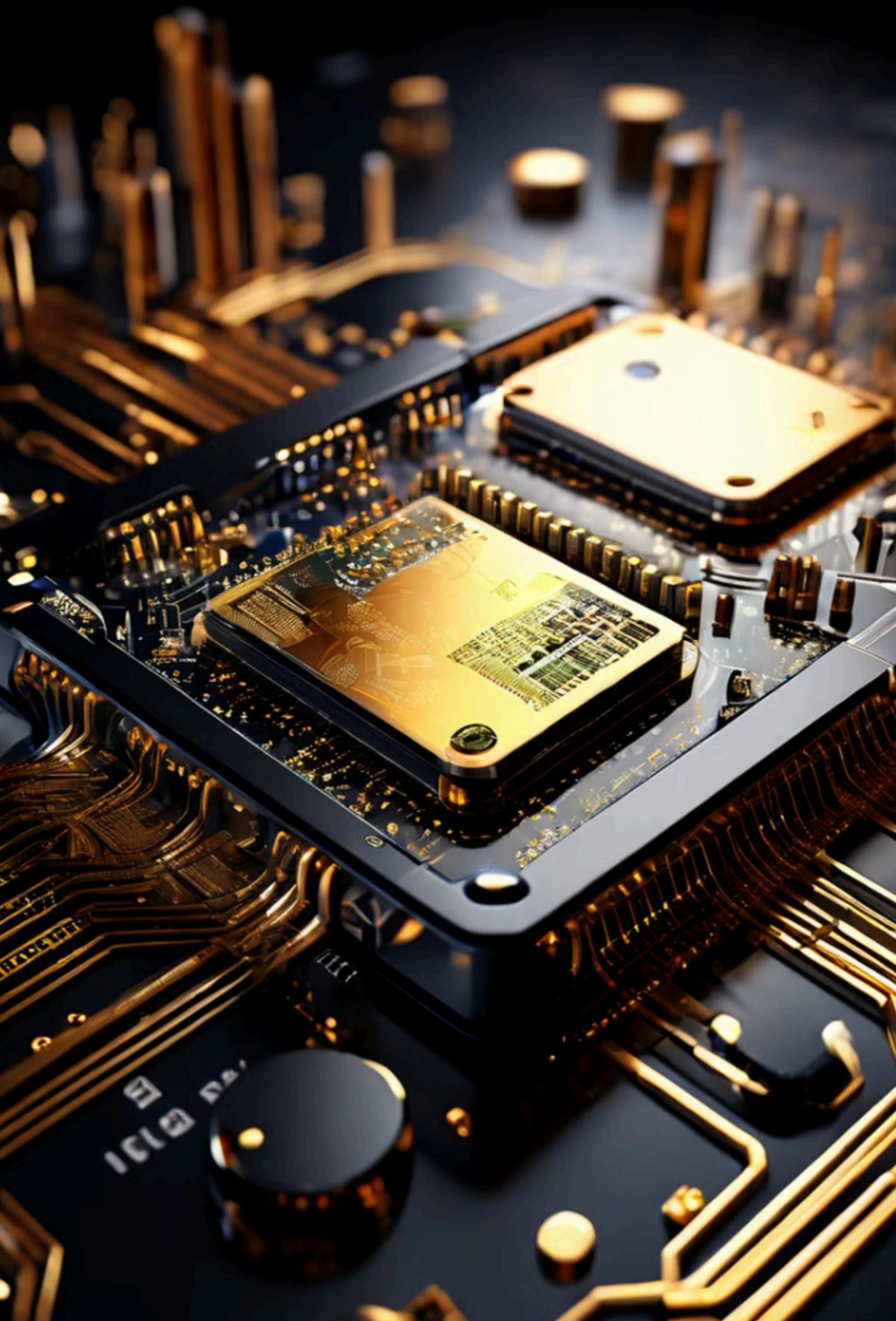
GRUPO:4IM6

EQUIPO:

- FLORES RIVERA JUAN MANUEL
- MARTINEZ MACIEL EVAN ALEXANDER
- GUADARRAMA GONZALES IAN RODRIGO
- PINEDA SALGADO LUIS ANGLES
- DELGADO ANDRADE CRISTIAN ANTONIO
- HERNANDEZ TELLEZ HECTOR FIDEL

# Descripción y Control de Procesos

Los procesos son la unidad básica de trabajo en un sistema informático. Cada proceso se caracteriza por su código de programa, datos, pila, segmentos de texto y datos, y contexto del procesador. Los estados del proceso incluyen nuevo, listo, ejecución, esperando y terminado. El Bloque de Control de Proceso (PCB) es una estructura de datos que almacena información clave sobre cada proceso, como su estado, contador de programa, registros de CPU, información de memoria y más.



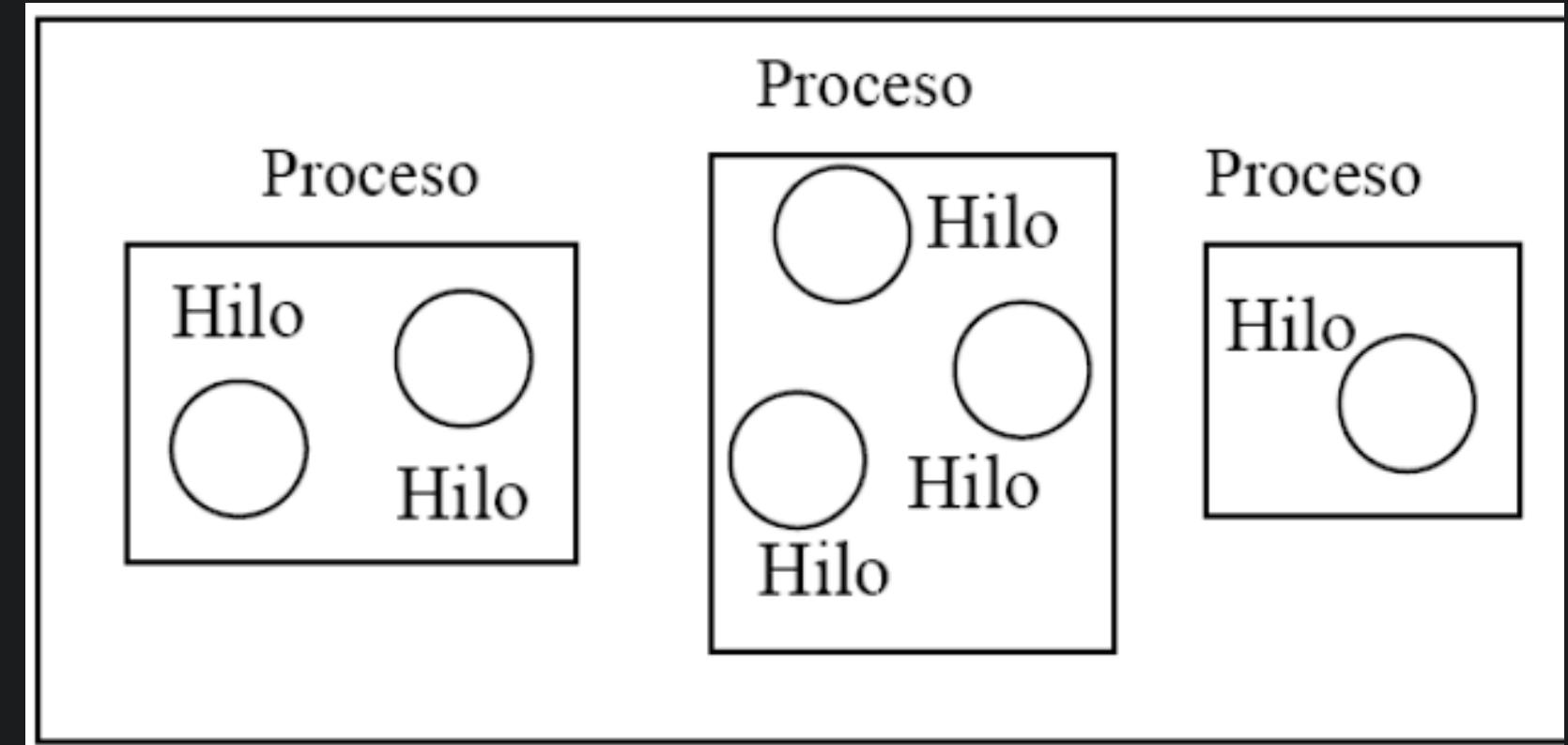
# Procesos y Hilos

## Procesos

Los procesos son instancias individuales de programas en ejecución, cada uno con su propio espacio de memoria independiente.

## Hilos

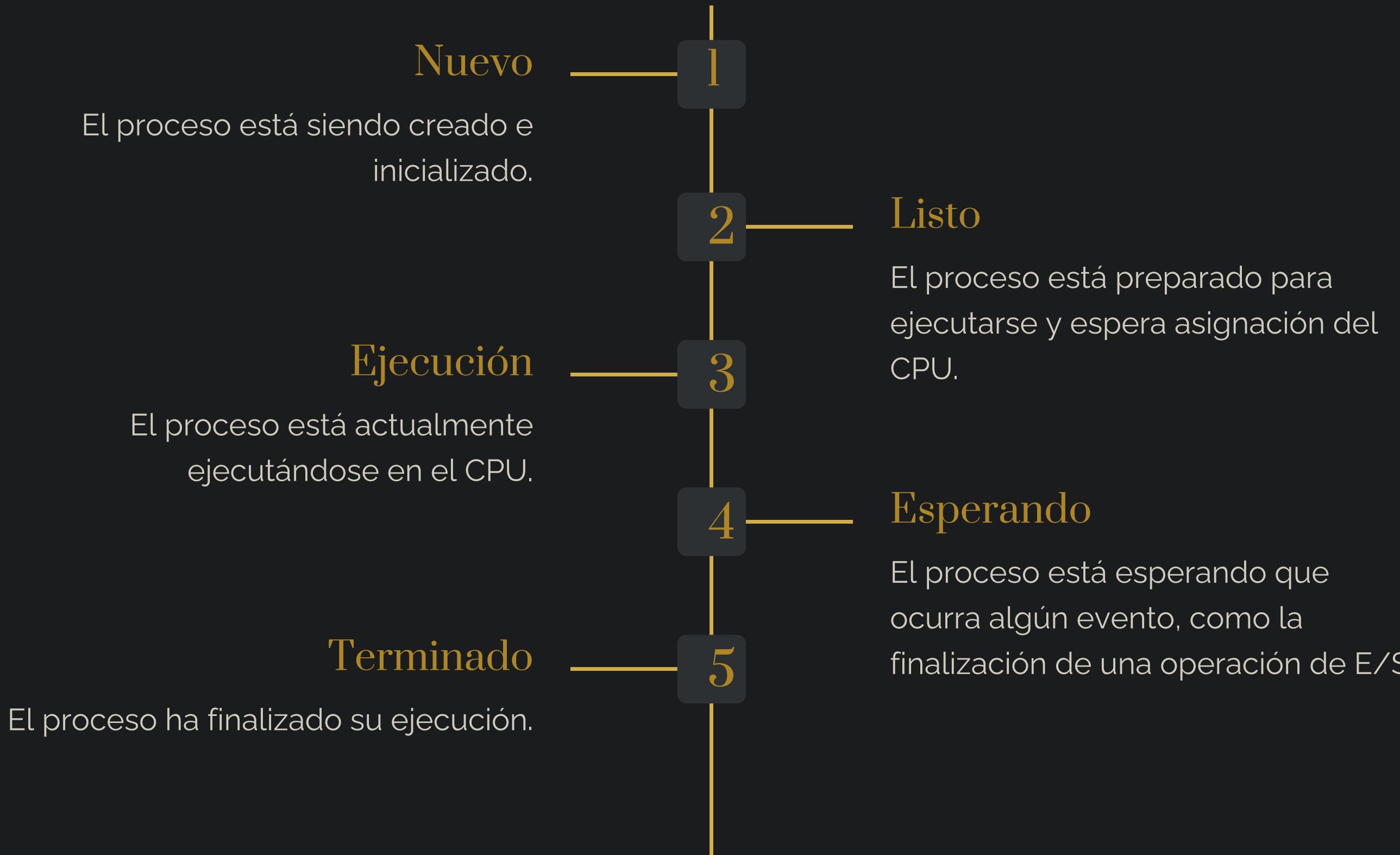
Los hilos son unidades de ejecución dentro de un proceso que comparten el mismo espacio de memoria y pueden ejecutarse concurrentemente.



## Multitarea

La multitarea permite a los sistemas operativos ejecutar múltiples procesos de forma aparentemente simultánea, utilizando técnicas como la commutación de contexto.

# Estados del Proceso



# Bloque de Control de Proceso (PCB)

1

## Información Clave

El PCB almacena toda la información relevante sobre un proceso, como su estado, contador de programa, registros de CPU, información de memoria, gestión de archivos, prioridad y ID del proceso (PID).

2

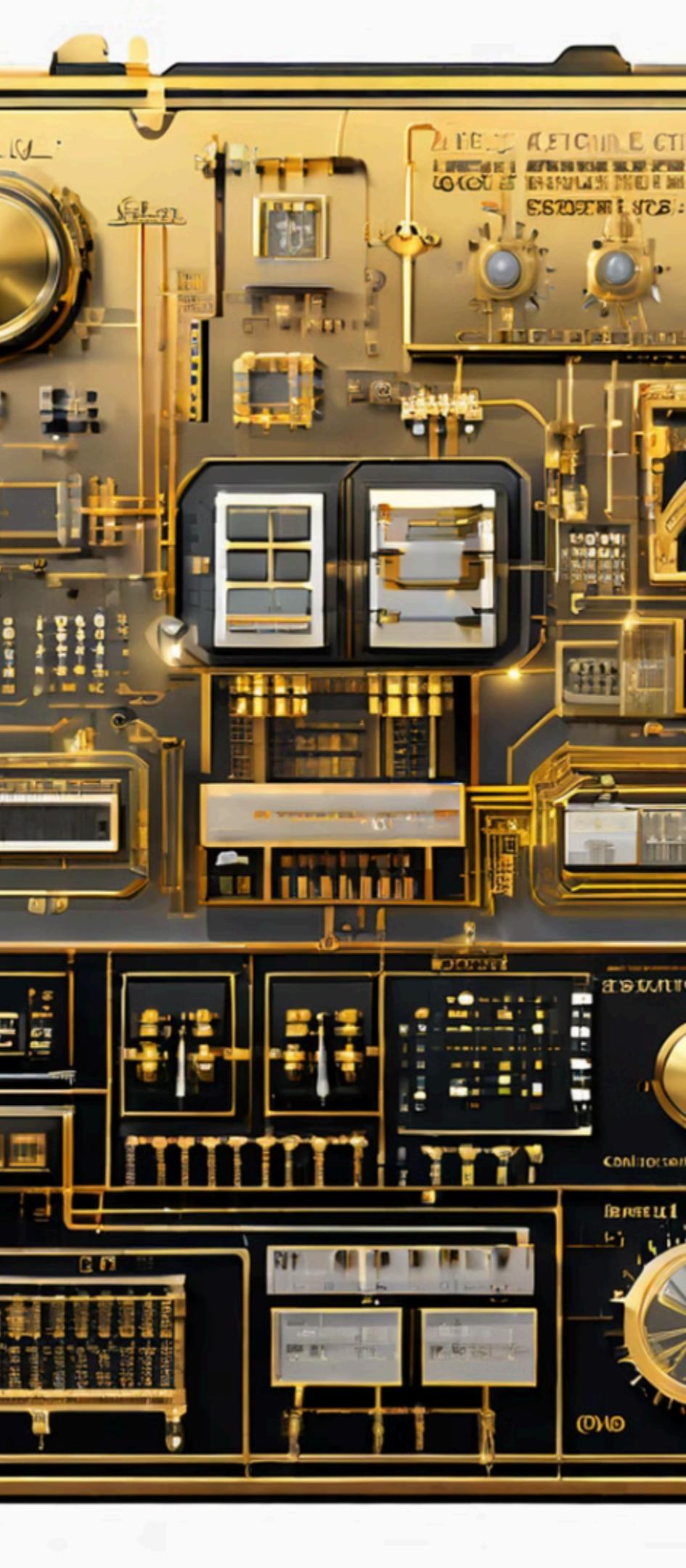
## Estructura de Datos

El PCB es una estructura de datos fundamental para el sistema operativo, ya que le permite mantener un control detallado y eficiente de cada proceso en ejecución.

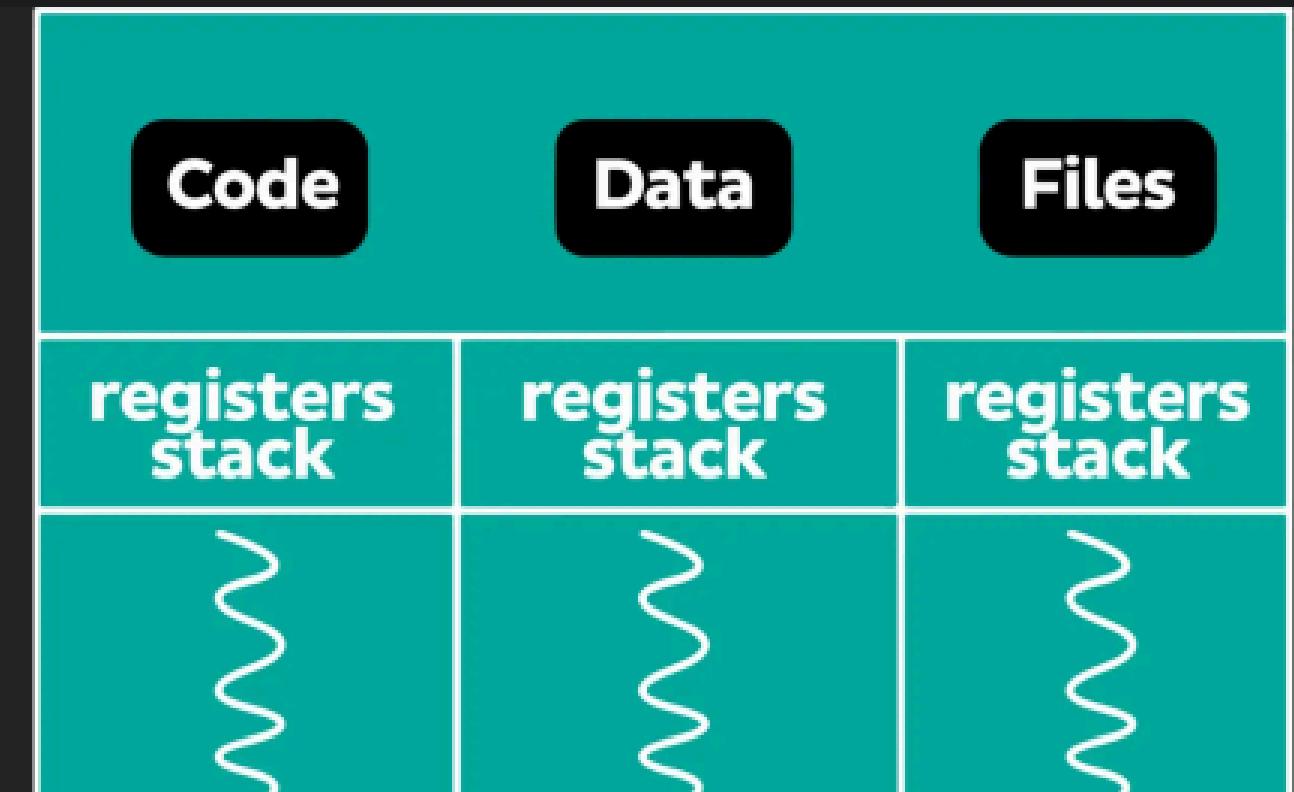
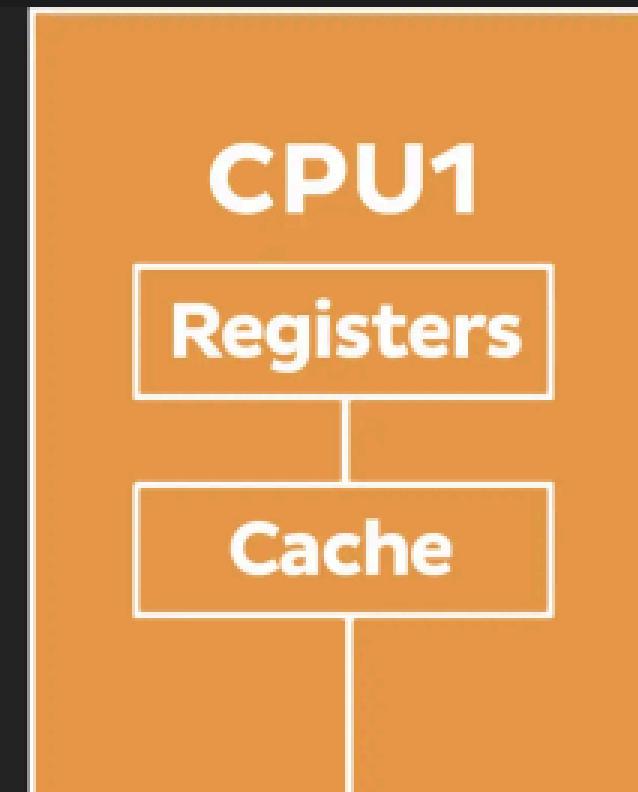
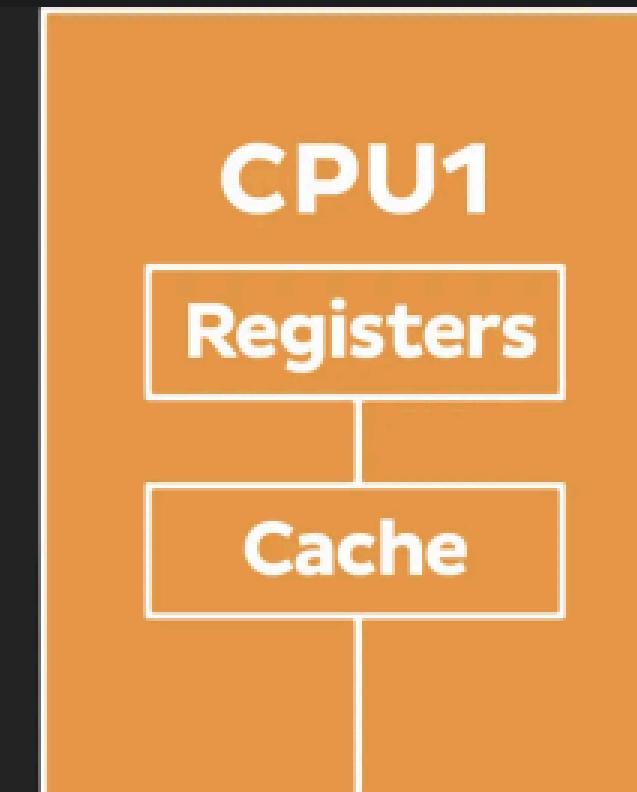
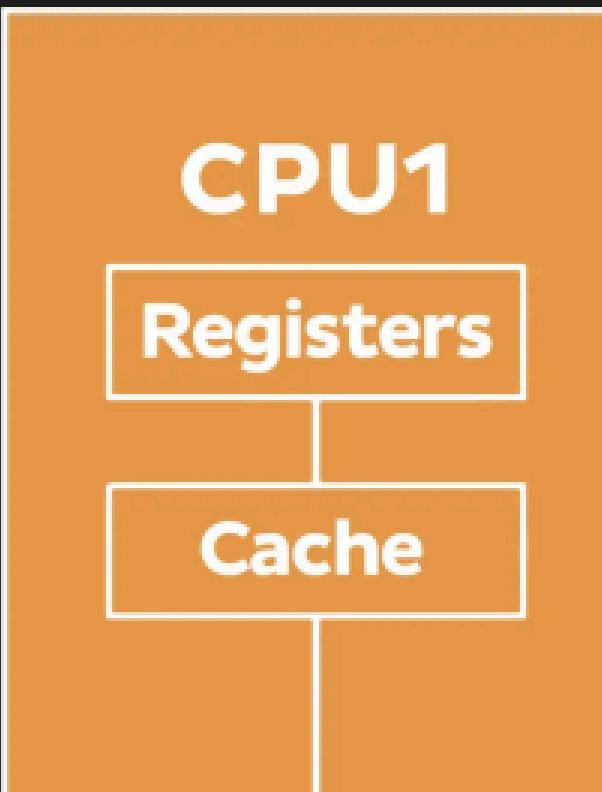
3

## Importancia para la Gestión

La información contenida en el PCB es crucial para que el sistema operativo pueda programar, planificar y administrar correctamente la ejecución de los procesos.



# Multitarea vs. Multithreading



## Multitarea

La multitarea permite a los sistemas operativos ejecutar múltiples procesos de forma aparentemente simultánea, utilizando técnicas como la conmutación de contexto.

## Multithreading

El multithreading permite la ejecución paralela de diferentes hilos dentro del mismo proceso, facilitando el rendimiento en sistemas con múltiples núcleos de CPU.

## Diferencias Clave

Mientras que la multitarea se refiere a la ejecución de múltiples procesos independientes, el multithreading se centra en la ejecución paralela de hilos dentro de un mismo proceso.



# Beneficios del Multithreading

1

## Mayor Eficiencia

El multithreading permite aprovechar mejor los recursos del sistema, especialmente en procesadores con múltiples núcleos, ya que los hilos pueden ejecutarse en paralelo.

2

## Mejor Respuesta

Los hilos pueden ejecutar tareas específicas de manera más rápida y eficiente, lo que mejora la capacidad de respuesta general del sistema.

3

## Optimización de Recursos

Al compartir el mismo espacio de memoria, los hilos pueden intercambiar datos y coordinar sus actividades de manera más fluida que los procesos independientes.

# Aplicaciones del Multithreading



## Navegadores Web

Los navegadores web utilizan múltiples hilos para manejar simultáneamente la carga de páginas, renderizado, descarga de recursos y procesamiento de scripts.



## Editores de Video

Los editores de video aprovechan el multithreading para procesar efectos, transiciones y renderizado de video de manera paralela.



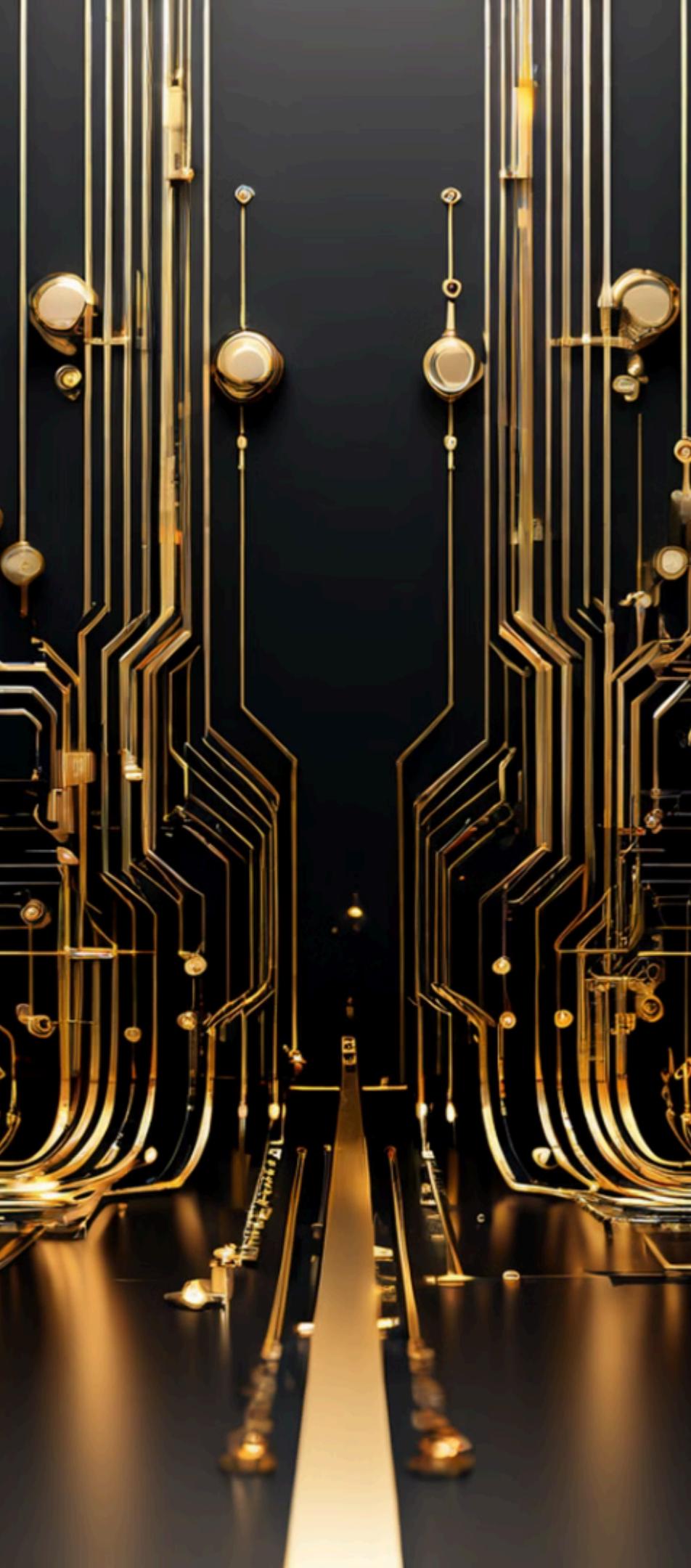
## Entornos de Desarrollo

Los entornos de desarrollo integrado (IDE) utilizan hilos para realizar tareas en segundo plano, como compilación, depuración y análisis de código.



## Servidores de Bases de Datos

Los servidores de bases de datos usan multithreading para manejar múltiples consultas y transacciones simultáneamente de manera eficiente.



# Consideraciones del Multithreading

1

## Sincronización

Los hilos deben sincronizar el acceso a recursos compartidos para evitar problemas como condiciones de carrera y bloqueos.

2

## Planificación

El sistema operativo debe planificar cuidadosamente la ejecución de los hilos para maximizar el rendimiento y evitar problemas de rendimiento.

3

## Escalabilidad

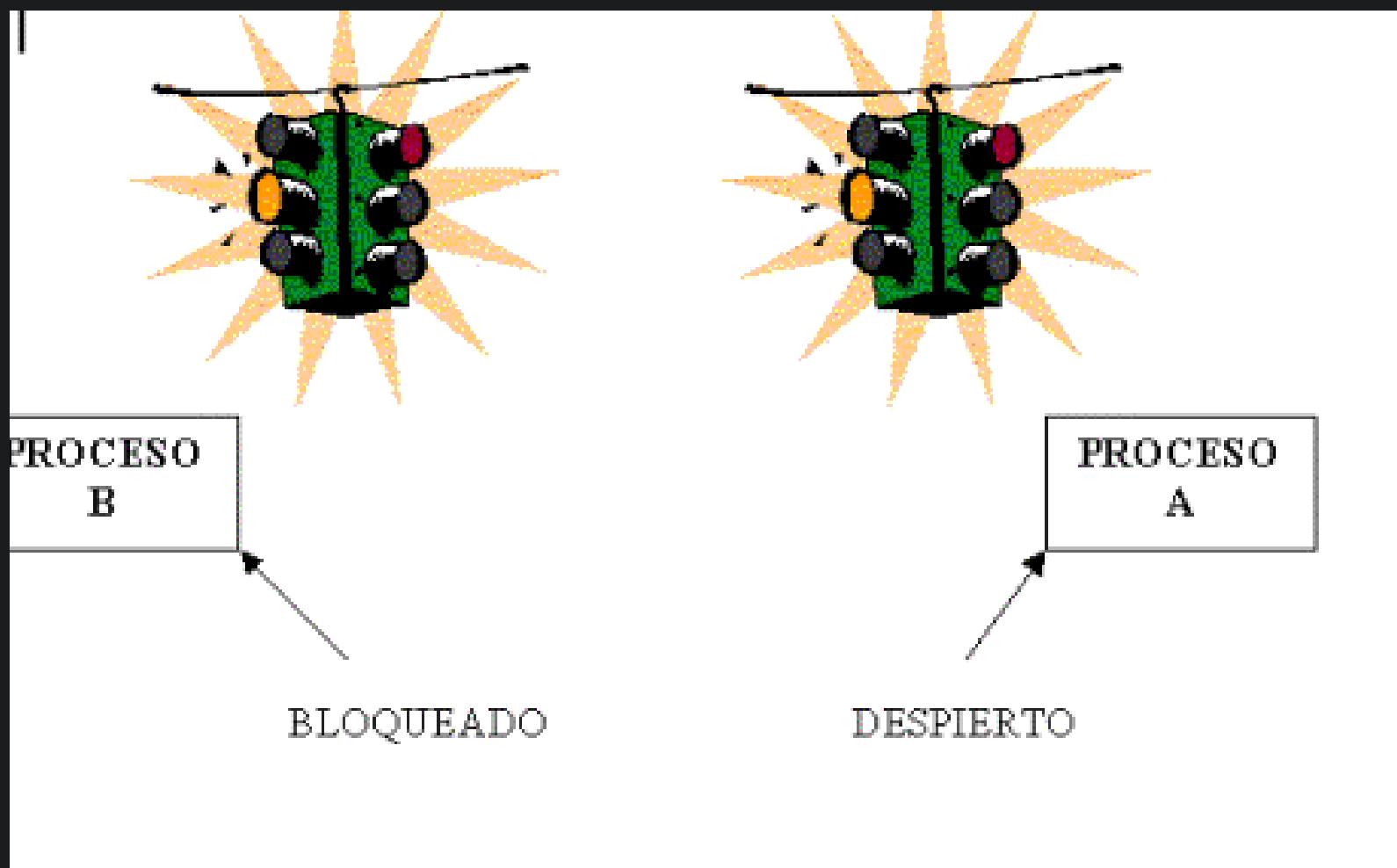
El número de hilos debe equilibrarse para aprovechar al máximo los recursos del sistema sin sobrecargarlo.

# Sincronización y Comunicación de Procesos

Los sistemas operativos modernos se basan en la concurrencia y la coordinación de múltiples procesos que deben interactuar de manera eficiente. En este contexto, la sincronización y comunicación de procesos se vuelven fundamentales para evitar problemas como condiciones de carrera y bloqueos. En las siguientes secciones, exploraremos los principales mecanismos utilizados para lograr esta sincronización y comunicación, incluyendo semáforos y paso de mensajes.



# Semáforos



## Semáforos Binarios

Los semáforos binarios tienen solo dos estados, 0 y 1, y se utilizan principalmente para implementar exclusión mutua (mutex), permitiendo que solo un proceso acceda a un recurso compartido a la vez.

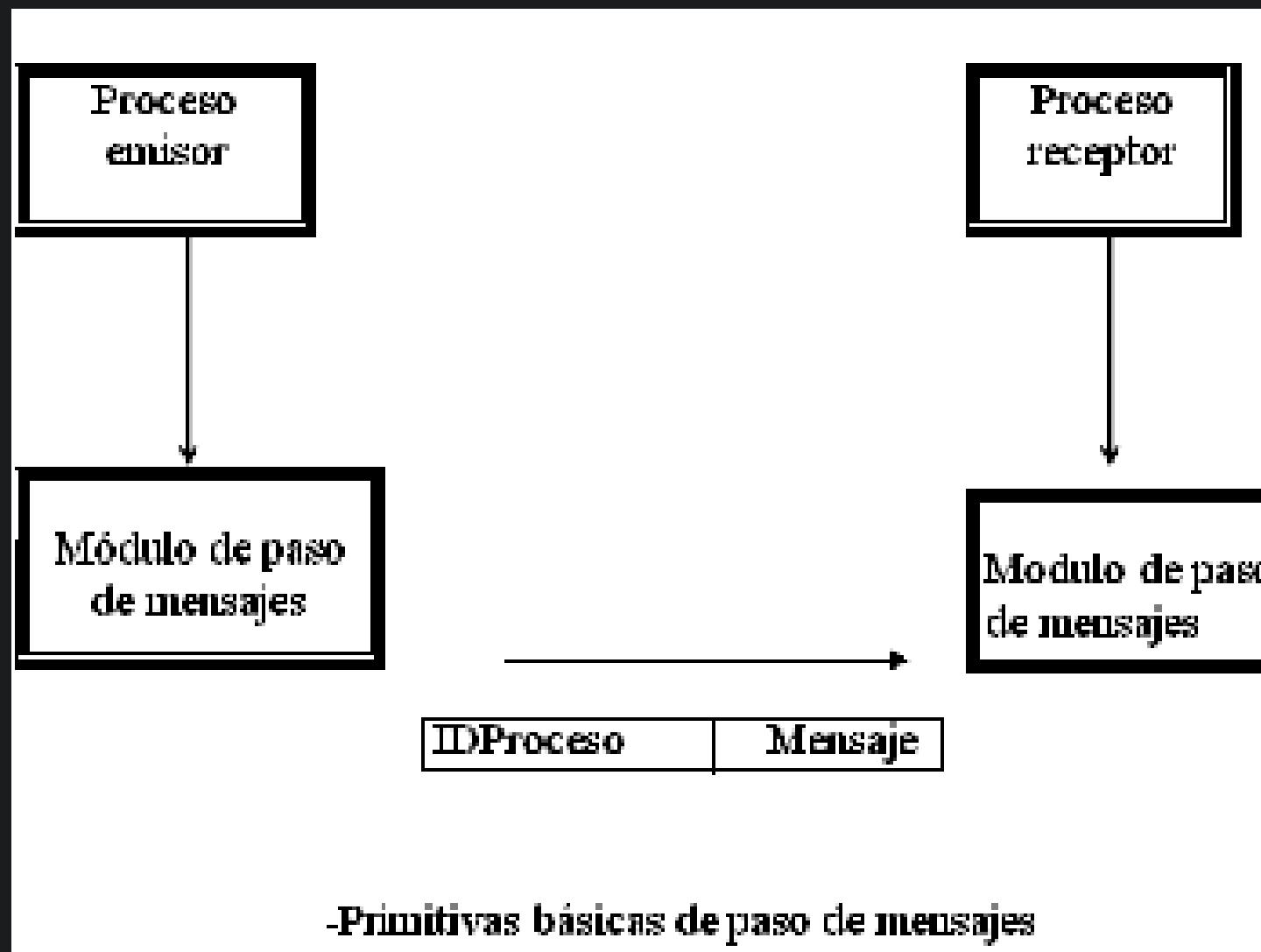
## Operaciones Básicas

Las operaciones básicas sobre semáforos son wait (P), que decrementa el valor del semáforo, y signal (V), que lo incrementa. Estas operaciones permiten a los procesos sincronizar su acceso a recursos compartidos.

## Semáforos Contadores

Los semáforos contadores permiten un valor mayor a 1 y se utilizan para controlar el acceso a un conjunto de recursos idénticos, como un grupo de impresoras o sockets de red.

# Paso de Mensajes



## Comunicación Síncrona y Asíncrona

La comunicación entre procesos mediante el paso de mensajes puede ser síncrona (bloqueante), donde el proceso emisor espera a que el receptor reciba el mensaje, o asíncrona (no bloqueante), donde el emisor no espera la recepción.

## Colas de Mensajes

Las colas de mensajes actúan como buffers donde los mensajes se almacenan temporalmente hasta que el receptor los procesa, facilitando la comunicación y aislando a los procesos.

## Beneficios del Paso de Mensajes

El paso de mensajes facilita la comunicación en sistemas distribuidos, reduce las condiciones de carrera y permite una estructura modular y fácil de mantener.

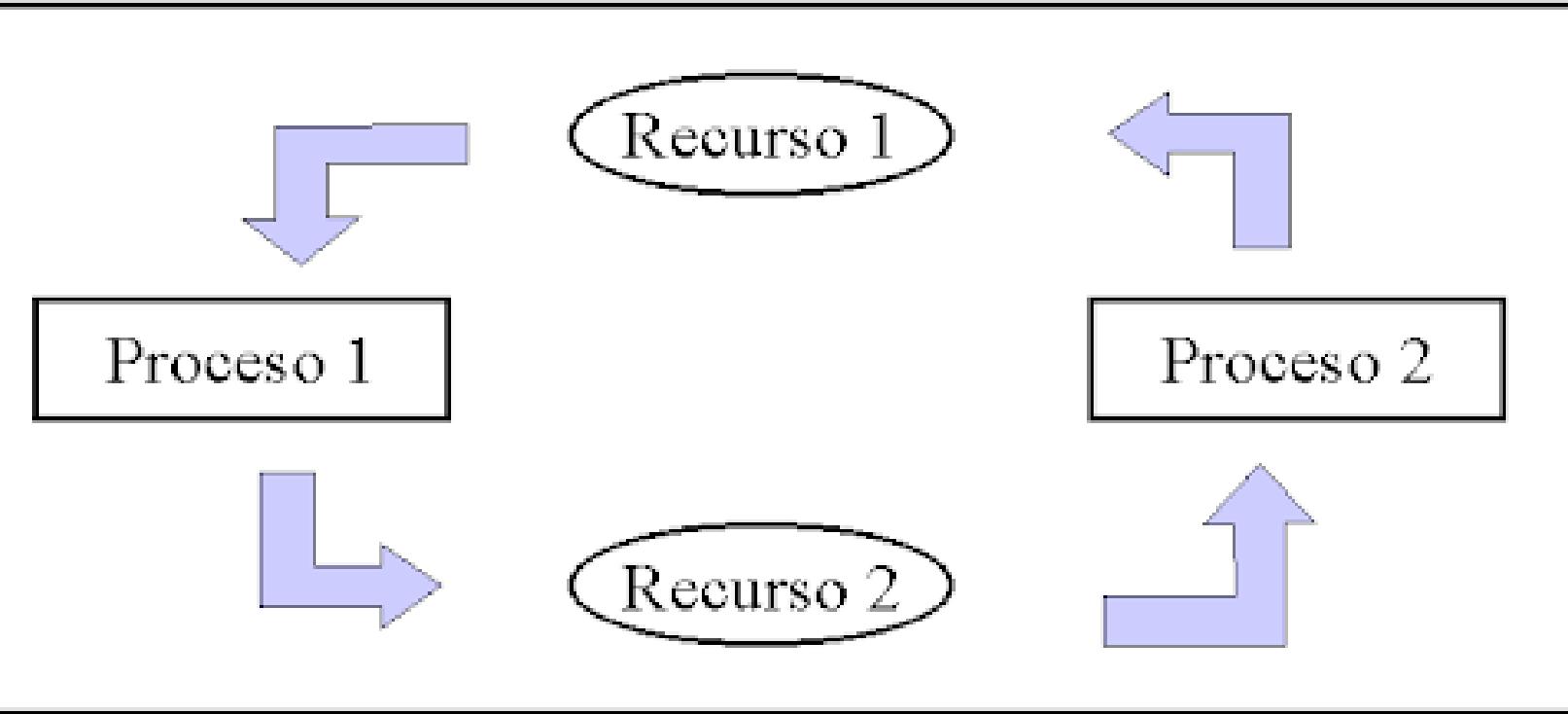
# Interbloqueo

## Condiciones de Interbloqueo

Las condiciones necesarias para que ocurra un interbloqueo (condiciones de Coffman) son: mutua exclusión, retención y espera, no apropiación y espera circular.

## Prevención del Interbloqueo

Para prevenir el interbloqueo, se pueden eliminar una de las condiciones necesarias, como mediante el protocolo de retención y espera o la asignación de recursos por adelantado.



## Evitación del Interbloqueo

Los algoritmos de evitación, como el algoritmo del Banquero, aseguran que el sistema no entre en un estado de interbloqueo al controlar las solicitudes de recursos.

## Detección y Recuperación

Cuando el interbloqueo ocurre, se pueden utilizar algoritmos de detección y técnicas de recuperación, como terminar procesos involucrados o reasignar recursos.

proceso



# Sincronización de Procesos

## Exclusión Mutua

La exclusión mutua es un mecanismo clave para garantizar que solo un proceso acceda a un recurso compartido a la vez, evitando condiciones de carrera.

## Sincronización de Acceso

Los semáforos permiten a los procesos sincronizar su acceso a recursos compartidos, controlando cuándo pueden acceder y cuándo deben esperar.

## Comunicación Efectiva

El paso de mensajes facilita la comunicación entre procesos, permitiendo el intercambio de datos sin necesidad de compartir memoria.

## Prevención de Bloqueos

Técnicas como la prevención y evitación del interbloqueo ayudan a evitar que el sistema se quede bloqueado en una situación de deadlock.

# Semáforos en Acción



## Inicialización

El semáforo se inicializa con un valor que representa la cantidad de recursos disponibles, como el número de impresoras o sockets de red.

## Solicitud de Acceso

Cuando un proceso necesita acceder a un recurso, llama a la operación wait (P) para decrementar el valor del semáforo.

## Concesión de Acceso

Si el valor del semáforo es mayor o igual a 1, el proceso obtiene acceso al recurso. Si no, el proceso se bloquea hasta que el semáforo esté disponible.

## Liberación de Recursos

Cuando el proceso termina de utilizar el recurso, llama a la operación signal (V) para incrementar el valor del semáforo y liberar el recurso.

# Comunicación entre Procesos



## Envío de Mensajes

Los procesos pueden enviar mensajes a través de una cola de mensajes, que actúa como un buffer temporal.



## Comunicación Asíncrona

El proceso emisor no tiene que esperar a que el receptor reciba el mensaje, lo que permite una comunicación más fluida.



## Escalabilidad

El uso de colas de mensajes facilita la comunicación en sistemas distribuidos, mejorando la escalabilidad y modularidad.



## Aislamiento de Procesos

La comunicación a través de mensajes aísla a los procesos, reduciendo las condiciones de carrera y facilitando el mantenimiento.