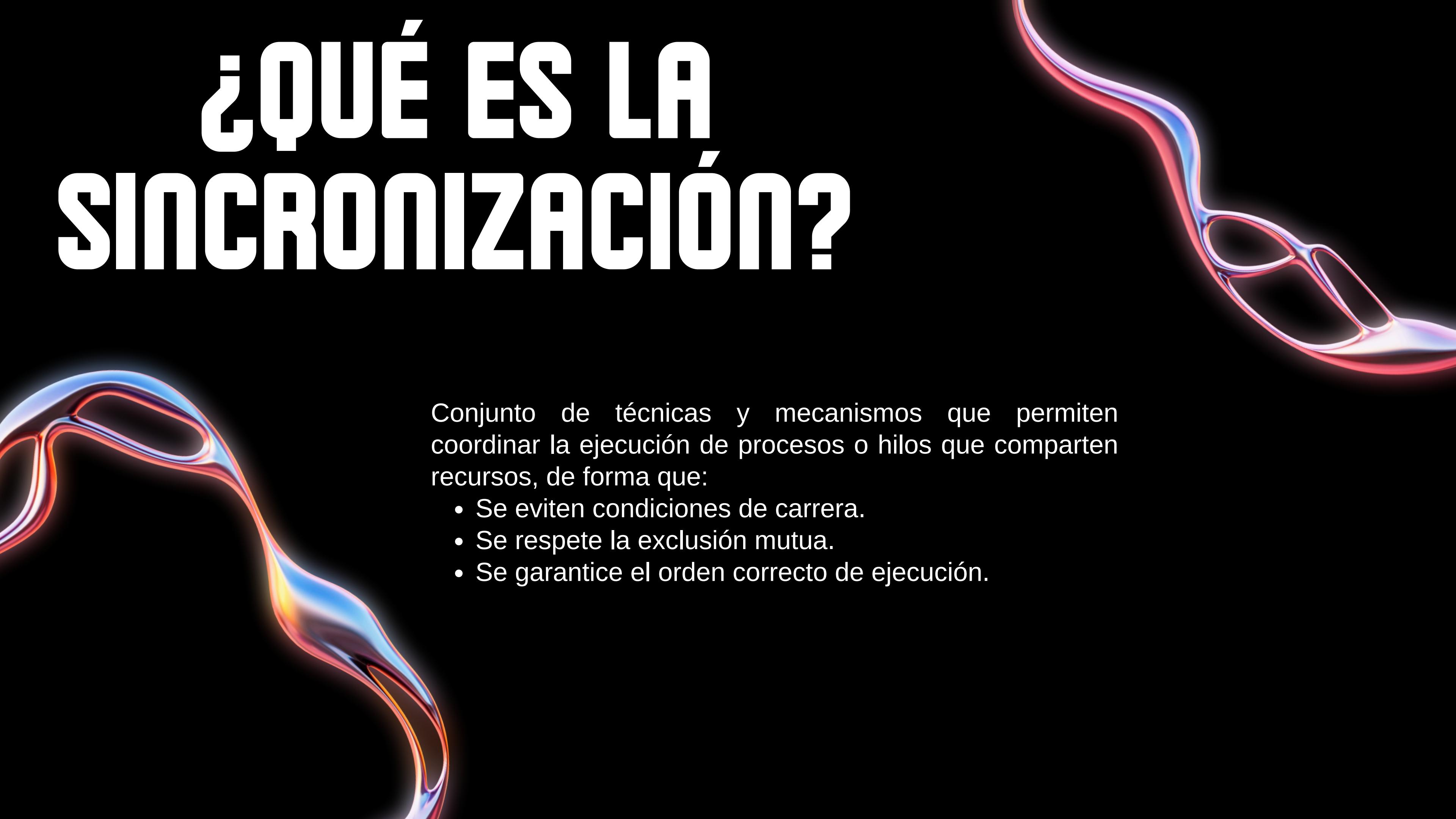


EXCLUSIÓN MUTUA

ALGORITMO DE SINCRONIZACIÓN

PRESENTA:
PEÑARRIETA VILLA JESUS

¿QUÉ ES LA SINCRONIZACIÓN?

The background features two abstract, glowing, wavy lines composed of multiple colored layers (red, orange, yellow, blue) against a black background. One line is positioned vertically along the left side, and another is positioned horizontally along the top right.

Conjunto de técnicas y mecanismos que permiten coordinar la ejecución de procesos o hilos que comparten recursos, de forma que:

- Se eviten condiciones de carrera.
- Se respete la exclusión mutua.
- Se garantice el orden correcto de ejecución.

SECCIÓN CRÍTICA

- Parte del programa donde un recurso compartido (variable, archivo, dispositivo, memoria) es accedido o modificado.
- Solo un proceso a la vez debe ejecutar esa sección, para evitar errores.



CONDICIONES DE CARRERA



- Ocurren cuando dos o más procesos/hilos acceden y modifican un recurso compartido(sección critica) al mismo tiempo.
- El resultado depende del orden de ejecución, lo que genera comportamientos impredecibles o errores.
- Ejemplo clásico: dos cajeros automáticos accediendo a la misma cuenta bancaria y retirando dinero sin coordinación.

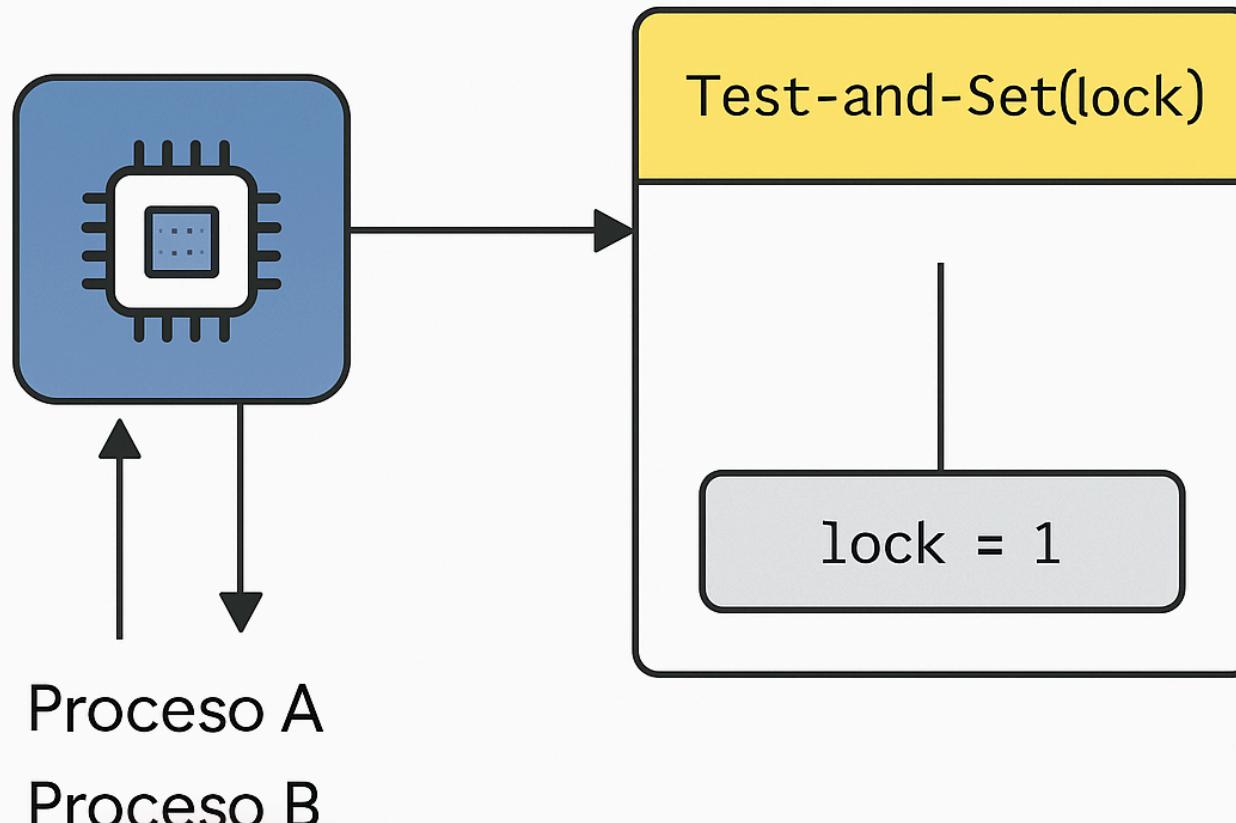
EXCLUSION MUTUA

La exclusión mutua es un mecanismo de sincronización que garantiza que solo un proceso o hilo pueda acceder a la sección crítica (un recurso compartido) en un momento dado.

SOLUCION HARDWARE

La exclusión mutua a nivel de hardware se basa en instrucciones atómicas y deshabilitación de interrupciones, siendo la base sobre la que se construyen muchos mecanismos de software modernos (como semáforos y monitores).

Exclusión Mutua a Nivel de Hardware



SOLUCIÓN SOFTWARE



Se utilizan algoritmos que garantizan que un solo proceso acceda a la sección crítica en cada momento, sin necesidad de instrucciones especiales de hardware.

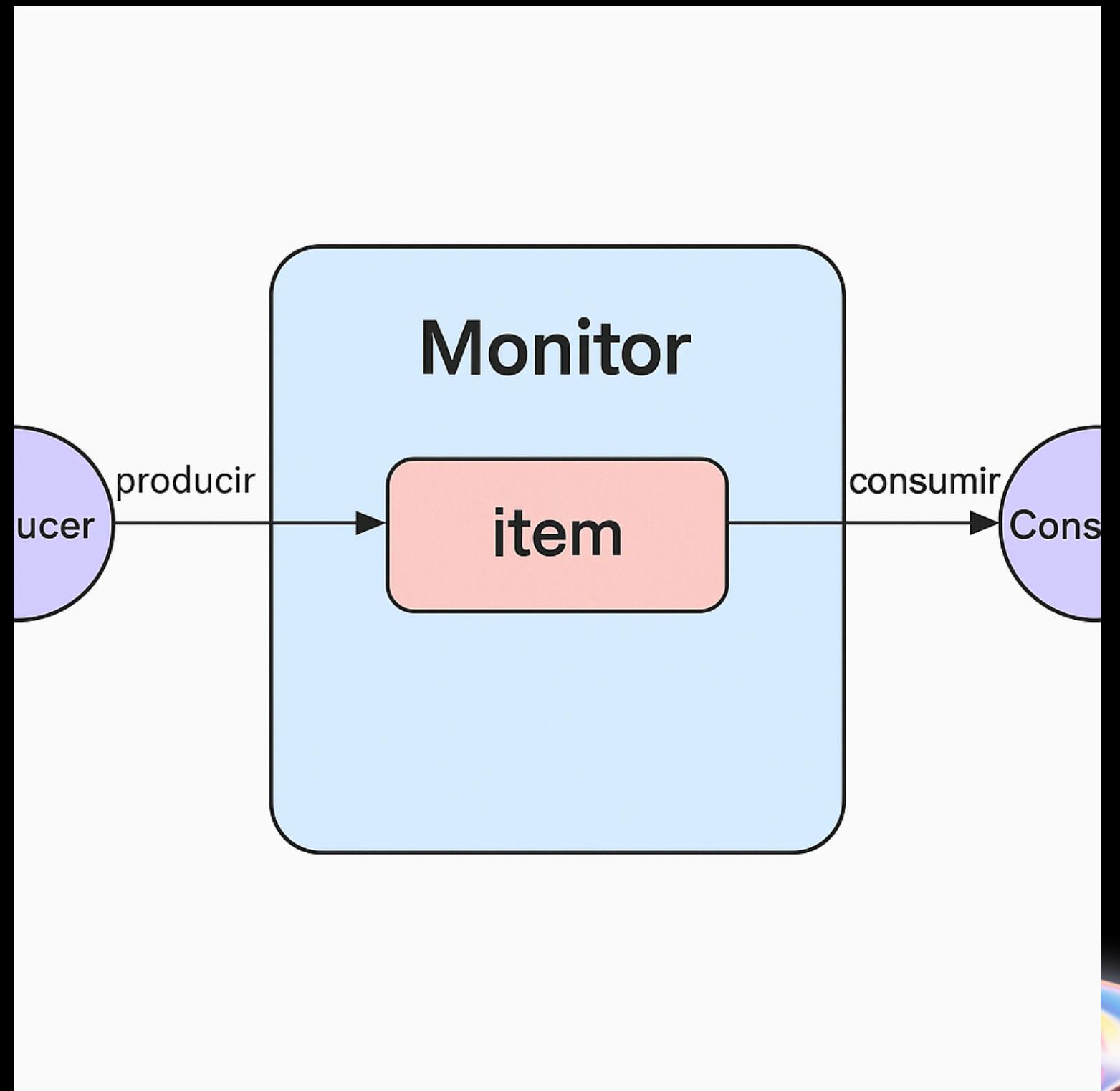
- El algoritmo de Dekker
- El algoritmo de Peterson
- algoritmo de panadería de Lamport
- El algoritmo de Szymański
- algoritmo de panadería blanco-negro de Taubenfeld
- El algoritmo de Maekawa

EXCLUSIÓN MUTUA RECUPERABLE

La mayoría de los algoritmos clásicos asumen que no habrá fallos durante la sección crítica.

- En la práctica, pueden ocurrir:
 - Cortes de energía.
 - Errores en hardware o interconexiones.
 - Procesos bloqueados o caídos.
 - Si un proceso falla dentro de la sección crítica, las soluciones convencionales pueden quedar bloqueadas o romper sus propiedades de seguridad.
 - La exclusión mutua recuperable busca mantener la seguridad y el progreso incluso ante fallos.
- Locks (mutexes)
 - Lectores – bloquea el escritor
 - Cerraduras recuperadas
 - Semaphores
 - Monitores
 - Mensaje que pasa
 - Tuple space

MONTORES



Propuesto por C.A.R. Hoare (1974) y perfeccionado por Per Brinch Hansen. Un monitor es una abstracción de alto nivel que combina:

- Datos compartidos.
- Procedimientos que los manipulan.
- Mecanismos de sincronización automáticos.
- Solo un proceso/hilo puede ejecutar código dentro del monitor a la vez.

CARACTERÍSTICAS PRINCIPALES

1. Exclusión automática:

- El lenguaje/sistema garantiza que solo un hilo entra al monitor.

2. Variables de condición:

- Se usan para esperar o notificar eventos dentro del monitor.
- Operaciones clave:
 - `wait(c)`: suspende al proceso hasta que otra señal llegue.
 - `signal(c)`: despierta a un proceso esperando en la condición `c`.

3. Mayor seguridad que los semáforos:

- Menos propensos a errores de programación.

VENTAJAS

1. Más fácil de usar que los semáforos, ya que encapsula la lógica de sincronización.
2. Reduce errores comunes como olvidarse de hacer signal.
3. Integrado en muchos lenguajes (Java: synchronized, C#: lock).

DESVENTAJAS

1. Menor flexibilidad que los semáforos en casos muy específicos.
2. Dependen del soporte del lenguaje o del sistema operativo.

GRACIAS