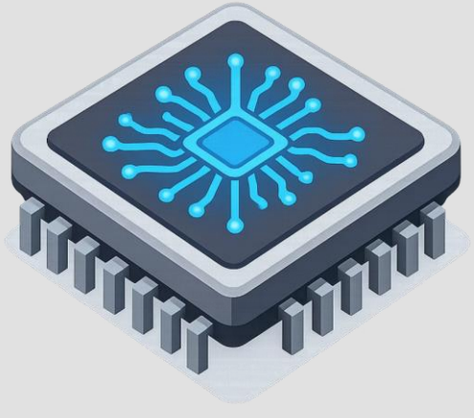
Unidad 4: Procesos e Hilos

El Corazón de la Multitarea



Unidad 4: Procesos

4.1. El concepto de proceso.

4.2. Planificación de procesos.

4.3. Operaciones con procesos.

4.4. Procesos cooperativos.

4.5. Hilos de Ejecución (Threads).

4.6. Comunicación entre procesos

Exploramos cómo los sistemas operativos gestionan la ejecución simultánea de programas, desde el concepto de proceso hasta la comunicación entre ellos.

El Concepto de Proceso

Un **proceso** es un programa en ejecución con su propio espacio de memoria y recursos asignados por el SO.
Es la unidad fundamental de trabajo del sistema.

🔍 **Analogía**: Un proceso es como un chef trabajando en su propia estación: tiene sus ingredientes, utensilios y avanza en su receta de forma independiente.

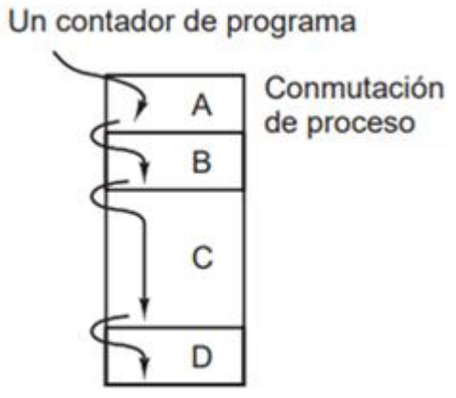
PID	**Estado**	**Contexto**
Identificador único	Listo · Ejecutando · Bloqueado	PC, registros, pila

Componentes clave

Concepto de Proceso

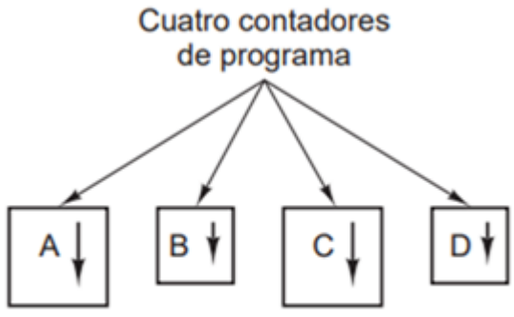
- Un proceso es:
  - Sección de texto (código del programa)
  - Actividad actual, representada por:
    - Valor del contador de programa.
    - Contenido de registros del procesador.
  - Además, también incluye:
    - Pila (stack), que contiene datos temporales (parámetros de subrutinas, direcciones de retorno y variables locales).
    - Sección de datos, que contiene variables globales y memoria dinámica.

Modelo del Proceso



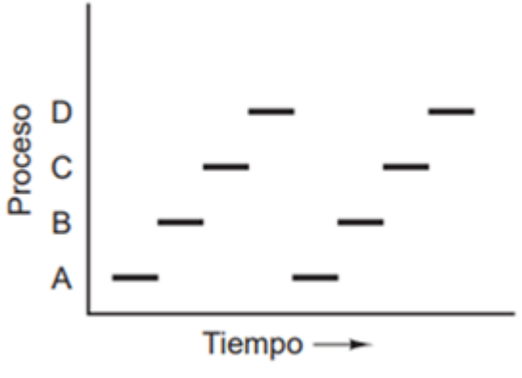
(a)

(a) Multiprogramación de cuatro programas.



(b)

(b) Modelo conceptual de cuatro procesos secuenciales independientes.



(c)

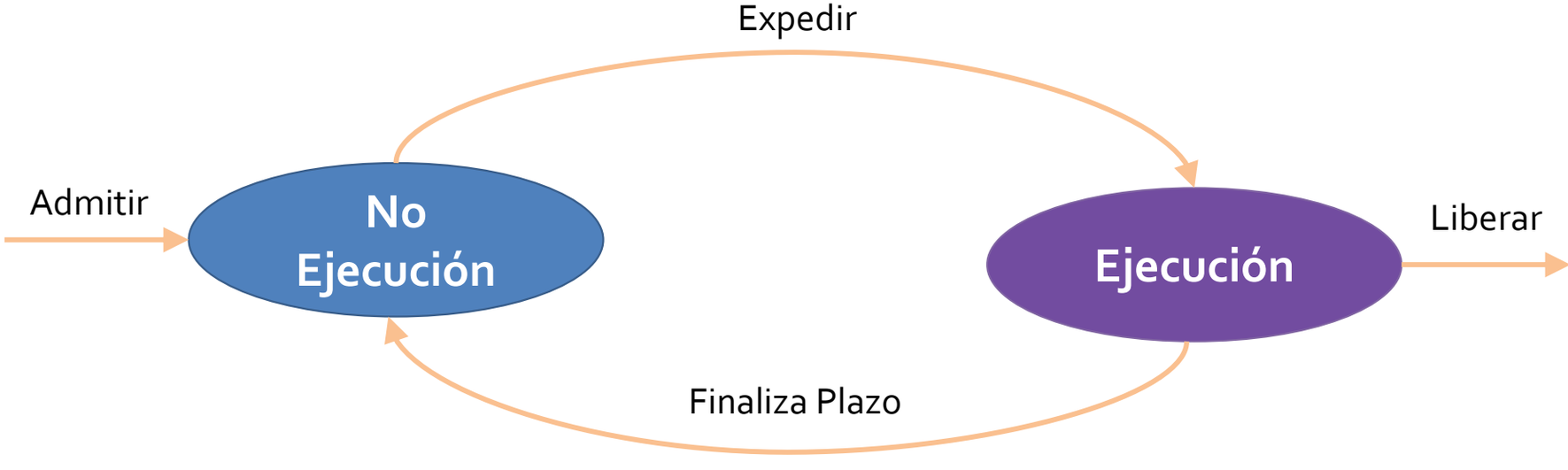
(c) Sólo hay un programa activo a la vez.

Fuente: Sistemas Operativos Modernos – Andrew Tanenbaum – 3° ed.

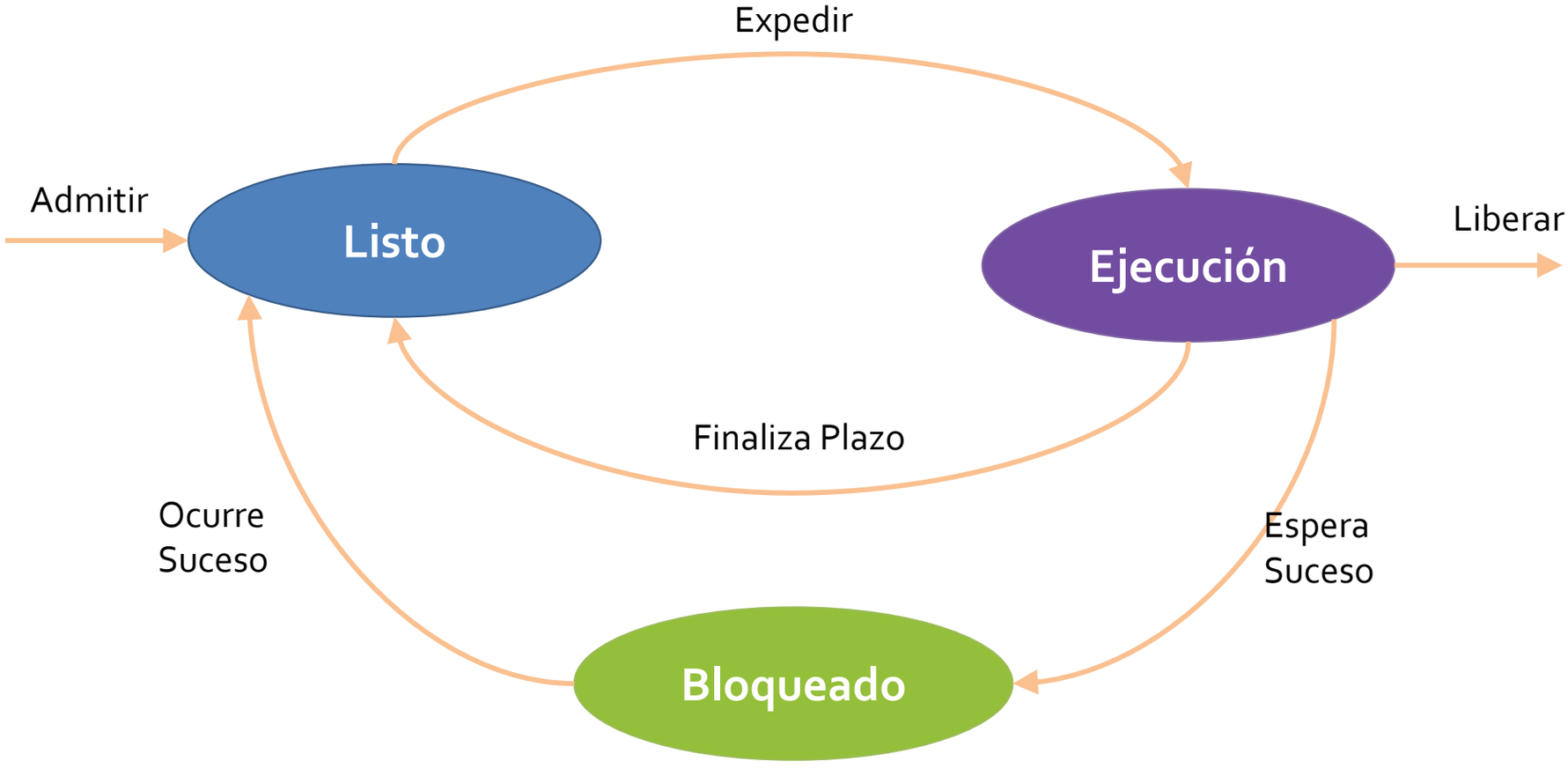
Estados de un Proceso

Un proceso no es simplemente un programa en ejecución, sino una entidad activa que cambia de estado a medida que el sistema operativo gestiona los recursos de la CPU.

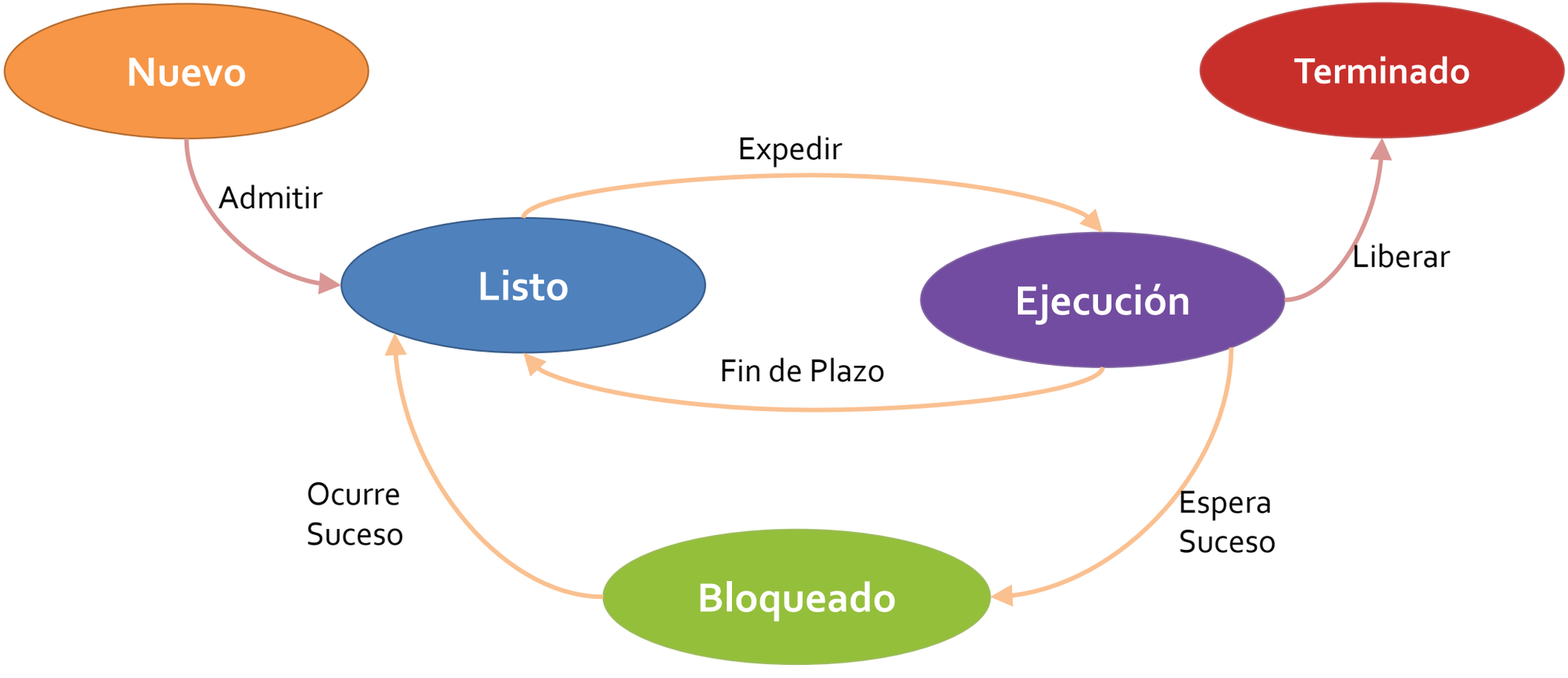
Modelo de 2 estados



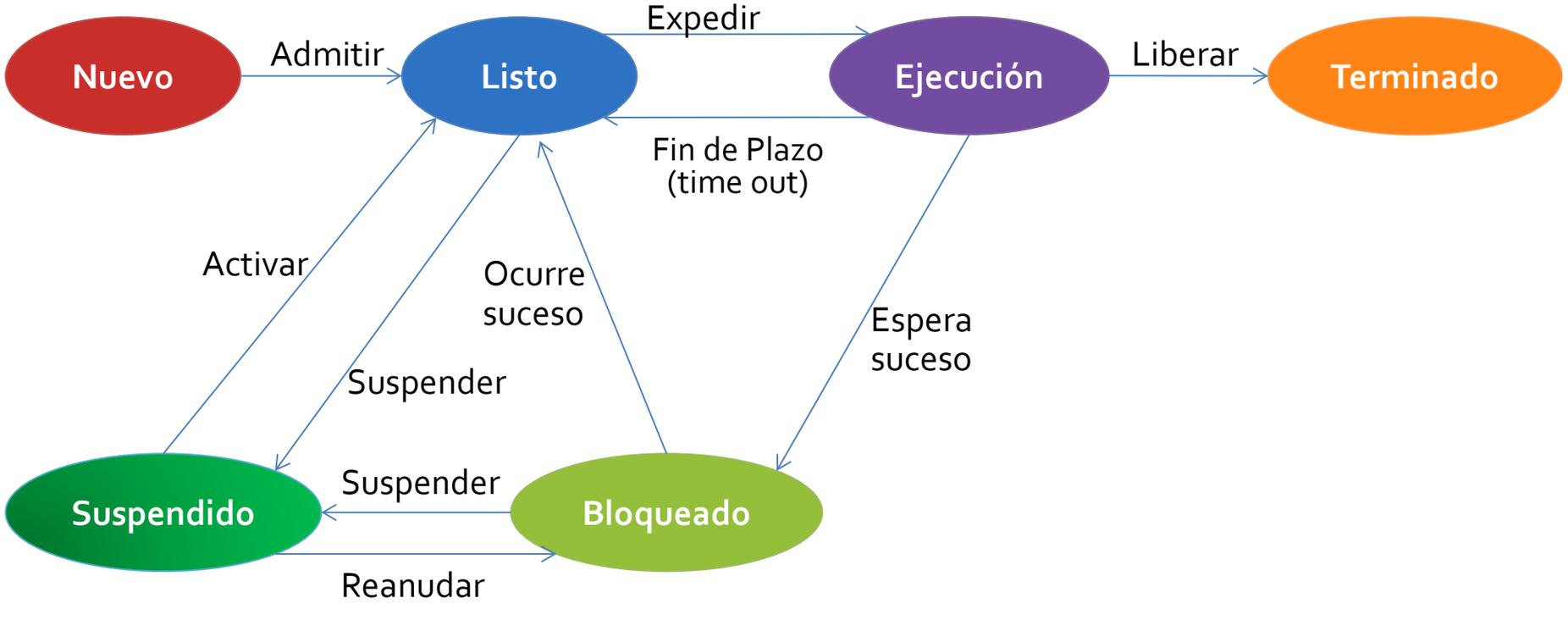
Modelo de 3 estados



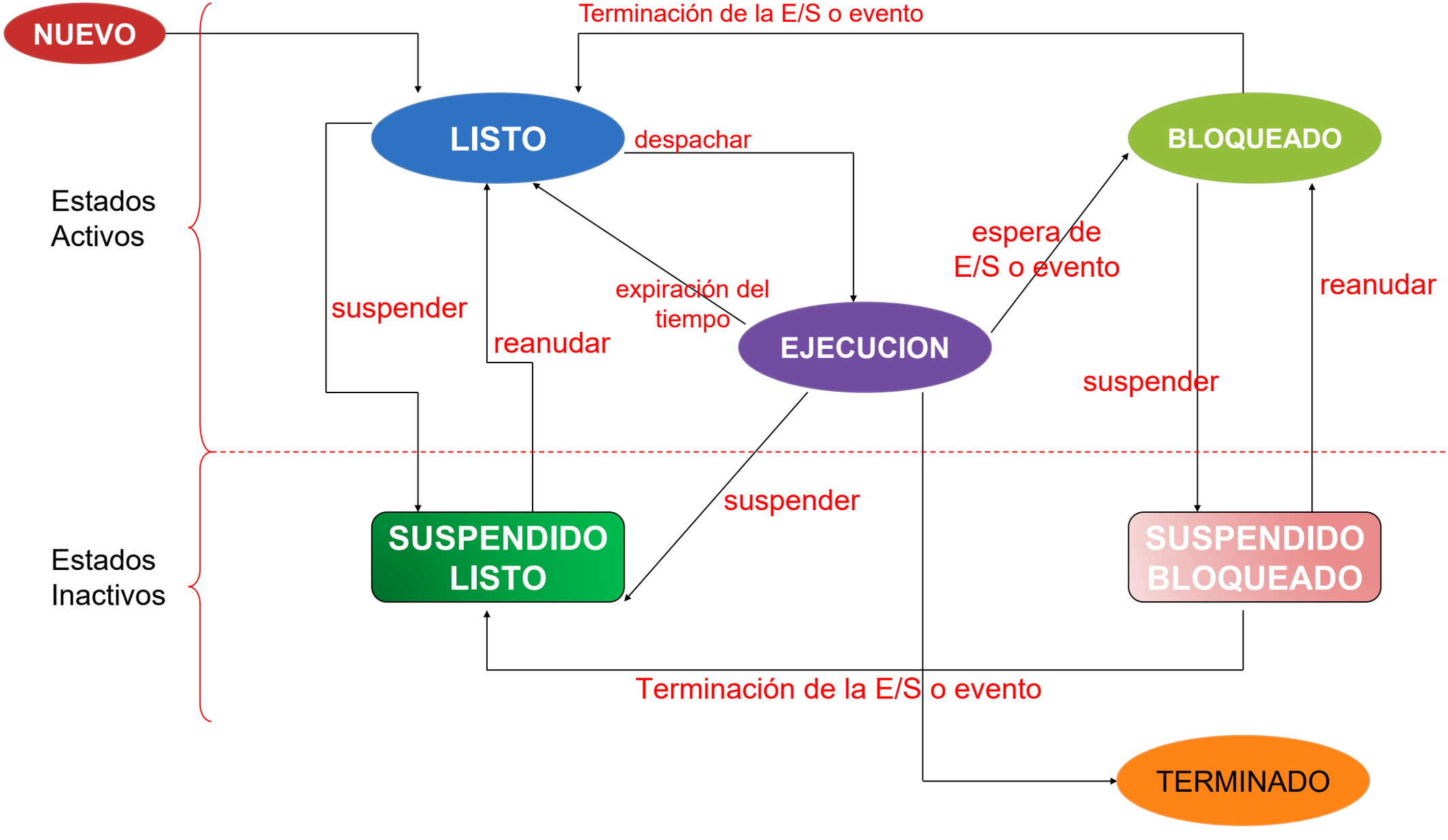
Modelo de 5 estados



Modelo de 6 estados



Modelo de 7 estados



Niveles De Planificación

- **Planificación a largo plazo – planificador de trabajos**
  - Decide cual será el próximo trabajo que se va a ejecutar, existe en los sistemas de proceso por lotes. Este nivel es el encargado de **crear** los procesos.
- **Planificación a medio plazo – planificador de Swapping**
  - Es el que decide si un proceso que esta en **ejecución**, **bloqueado** o **suspendido** debe ser extraído de la memoria temporalmente, posteriormente cuando el sistema se encuentre más descargado devolverá dicho proceso a la memoria. Existe en los sistemas de tiempo compartido.
- **Planificación a corto plazo – planificador del procesador**
  - Es el encargado de decidir cómo y cuando tendrá acceso al procesador un proceso que está **listo** para utilizarlo, lleva a cabo las funciones de multiprogramación, estando siempre residente en memoria.

Planificación de Procesos

- El planificador (scheduler) decide que proceso ocupa la CPU en cada instante, creando la ilusión de multitarea

- Cada algoritmo tiene ventajas según el tipo de carga de trabajo: interactiva, por lotes o de tiempo real.

Cambio de contexto: Guardar el estado del proceso actual y restaurar el del siguiente. Tiene un costo en tiempo de CPU que los algoritmos intentan minimizar.

Operaciones con Procesos

Creación y Terminación

fork()

Duplica el proceso padre creando un hijo idéntico

exec()

Reemplaza la imagen del proceso por un nuevo programa

Terminación

Salida normal, por error o señal de otro proceso

Operaciones con Procesos

Fenómenos especiales

🧟 Proceso Zombie

Terminó pero su padre no leyó su estado de salida. Consume una entrada en la tabla de procesos.

👤 Proceso Huérfano

Su proceso padre terminó antes que él. Es adoptado por `init` (PID 1).

Procesos Cooperativos

No todos los procesos compiten: muchos **colaboran** para alcanzar un objetivo común, compartiendo información y coordinando sus acciones.

Servidores Web

Un proceso maestro crea hijos para atender cada solicitud HTTP de forma independiente y concurrente.

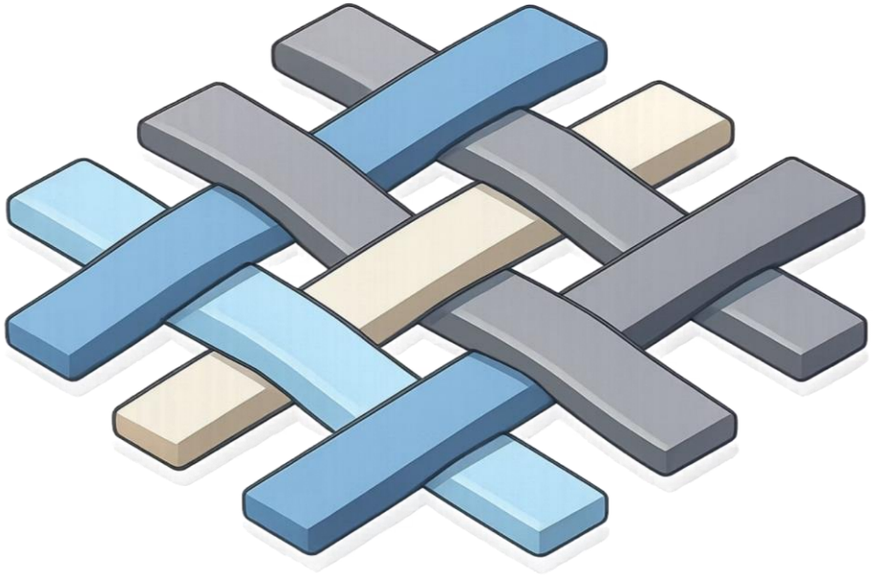
Sistemas Distribuidos

Procesos en distintas máquinas cooperan para procesar grandes volúmenes de datos (ej. MapReduce).

El Gran Desafío

La **sincronización** es crítica: sin ella, surgen condiciones de carrera e inconsistencia de datos.

Hilos de Ejecución (Threads)



Un **hilo** es un flujo de ejecución independiente dentro de un mismo proceso. Todos los hilos de un proceso comparten memoria y recursos.

👩‍🍳 **Analogía:** Los hilos son ayudantes del chef en la misma cocina — comparten ingredientes y utensilios, pero cada uno hace su tarea.

Ventajas clave:

- Menor sobrecarga que crear procesos completos
- Comunicación más rápida (memoria compartida)
- Mejor aprovechamiento de CPUs multinúcleo

Tipos de hilos

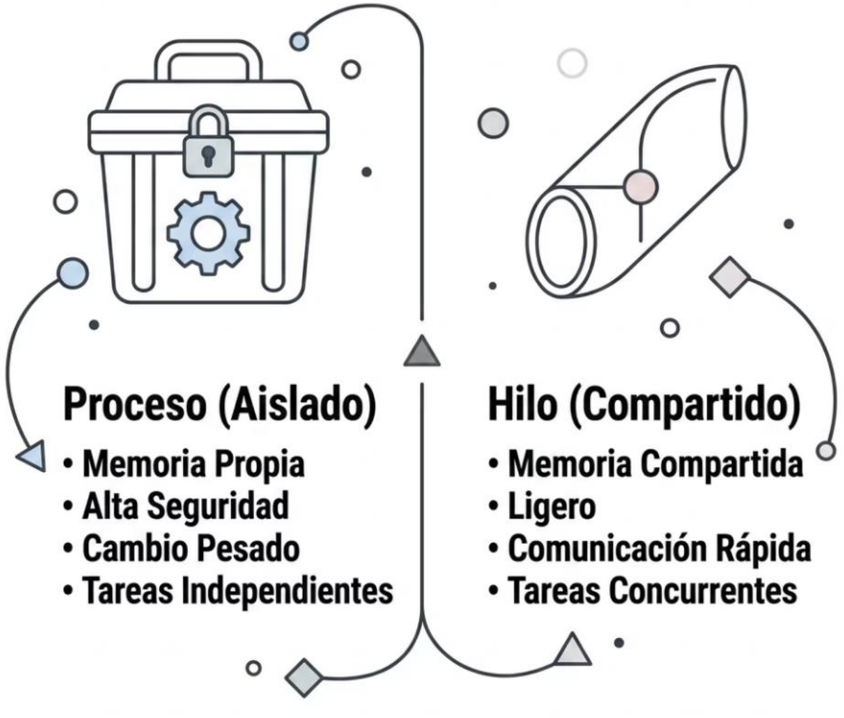
Usuario

Gestionados en espacio de usuario, sin soporte del kernel

Kernel

Gestionados directamente por el SO, mayor control

Hilo vs. Procesos: ¿Cuándo usar cada uno?



Ejemplo real: Navegador Web

Los navegadores modernos combinan ambos enfoques de forma inteligente:

→ **Procesos por pestaña**

Cada pestaña es un proceso independiente. Si una falla, las demás siguen funcionando.

→ **Hilos por tarea**

Dentro de cada pestaña, hilos separados descargan imágenes, renderizan HTML y ejecutan JavaScript concurrentemente.

Comunicación entre Procesos (IPC)

Los procesos independientes necesitan mecanismos para intercambiar datos y coordinarse. El SO provee varias formas de IPC:

Tuberías (Pipes) y FIFOs

Comunicación unidireccional entre procesos relacionados. Los FIFOs (named pipes) permiten comunicación entre procesos no relacionados.

Memoria Compartida

Región de memoria accesible por múltiples procesos. Es el método **más rápido,** pero requiere sincronización cuidadosa para evitar conflictos.

Paso de Mensajes

Los procesos se envían paquetes de datos estructurados mediante `send()` y `receive()`. Ideal para sistemas distribuidos.

UNIDAD 5

Planificación de Procesos

Unidad 5: Contenidos

5.1. Conceptos básicos.

5.2. Criterios de planificación.

5.3. Algoritmos de planificación.

5.4. Planificación en tiempo real.

5.5. Evaluación de algoritmos.

Cómo el sistema operativo decide qué se ejecuta, cuándo y por cuánto tiempo.

Conceptos Básicos: ¿Quién Manda en la CPU?

CPU

- El cerebro del ordenador. Solo puede ejecutar una instrucción a la vez.

MULTIPROGRAMACION

- La ilusión de ejecución simultánea: la CPU alterna entre procesos para maximizar su uso.

PROCESO

- Un programa en ejecución, con su propio estado, memoria y necesidades de CPU

PLANIFICADOR

- El director de orquesta del sistema operativo: decide qué proceso se ejecuta y cuándo.

Criterios de Planificación

¿Qué Buscamos en una Buena Planificación?

Utilización de la CPU

Mantener el procesador ocupado entre el **40% y 90%** del tiempo.

Rendimiento

Número de procesos completados por unidad de tiempo.

Tiempo de Espera

Tiempo que un proceso pasa en estado "listo" esperando la CPU.

Tiempo de Finalización

Tiempo total desde que un proceso inicia hasta que termina.

Tiempo de Respuesta

Desde la solicitud hasta la primera respuesta. **Crucial en sistemas interactivos.**

Algoritmos de Planificación: Las Reglas del Juego

FIFO

Primero en llegar, primero en ser atendido. Simple, pero vulnerable al **efecto convoy**.

SJF

El proceso con la ráfaga más corta se ejecuta primero. **Minimiza el tiempo de espera promedio**.

Round Robin

Cada proceso recibe un **quantum** de tiempo. Ideal para sistemas interactivos y time-sharing.

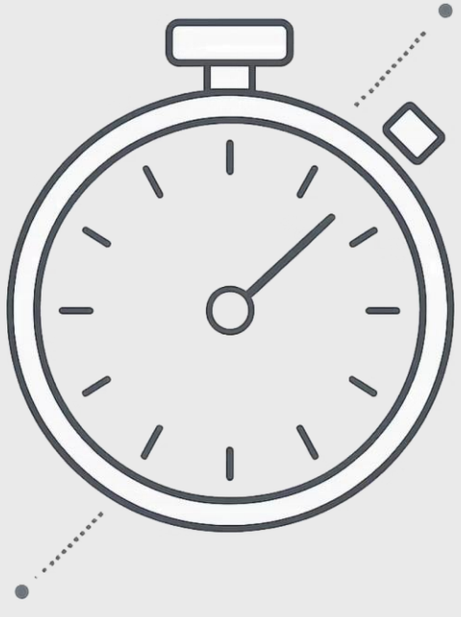
Prioridades

Los procesos VIP se ejecutan primero. Puede ser **fija** o **dinámica** según el sistema.

Flujo de los Algoritmos de Planificación

Cada algoritmo toma **decisiones distintas** ante la misma cola de procesos listos, generando comportamientos radicalmente diferentes en términos de equidad, eficiencia y tiempo de respuesta.

Planificación en Tiempo Real: ¡Cumplir Plazos es Vital!



⚠️ Tiempo Real Crítico

Incumplir un plazo es un **desastre**. Ej: control de vuelo, sistemas médicos, airbags.

⏱️ Tiempo Real No Crítico

Incumplir es indeseable, pero no catastrófico. Ej: streaming de video, videojuegos.

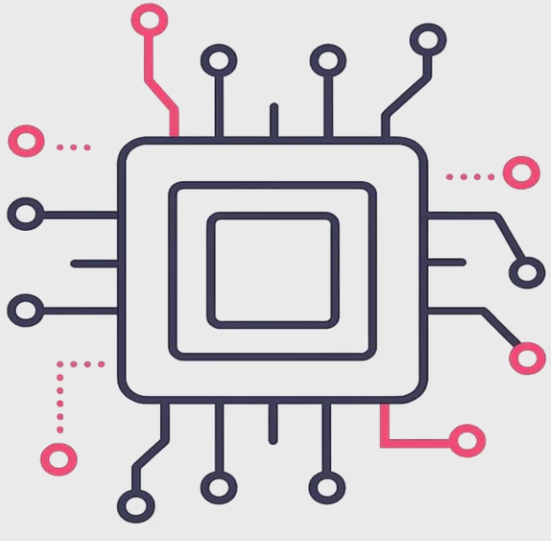
RMA — Rate Monotonic Analysis

Prioridades fijas: a **menor período**, mayor prioridad. Garantía matemática de cumplimiento.

EDF — Earliest Deadline First

Prioridades dinámicas: el proceso con la **fecha límite más cercana** se ejecuta primero.

Planificación en tiempo real. Planificación en Multiprocesadores



Los sistemas **multinúcleo** ofrecen mayor rendimiento, pero introducen nuevos desafíos de coordinación.

Contención de Recursos

Múltiples núcleos compiten por recursos compartidos como memoria y buses, generando interferencias.

Alojamiento de Tareas

Decidir en qué núcleo ejecutar cada tarea es clave para minimizar migraciones y maximizar la localidad.

Políticas Combinadas

Se buscan algoritmos que combinen estrategias para optimizar el uso de todos los núcleos simultáneamente.

Evaluación de Algoritmos

¿Cuál es el Mejor Algoritmo?

No existe un algoritmo **perfecto** para todas las situaciones. La elección depende de los objetivos del sistema:

→ **¿Rendimiento general o tiempo de respuesta?**

→ **¿Sistema interactivo o de tiempo real?**

→ **¿Monoprocesador o multiprocesador?**

Conclusión: El Arte de la Concurrencia

Procesos e hilos son los cimientos del software moderno. Dominar su gestión es esencial para cualquier ingeniero en informática.

01

Proceso = Programa en ejecución

Con PID, estado y contexto propio gestionado por el SO.

02

Planificación e IPC son clave

El scheduler y los mecanismos de comunicación determinan el rendimiento del sistema.

03

Hilos = Concurrencia eficiente

Menor costo que procesos, ideales para aprovechar CPUs multinúcleo.

04

Planificación de Procesos

Buenos tiempos de respuesta, mayor rendimiento del Sistema, justicia

🚀 **El futuro es concurrente:** Sistemas distribuidos, microservicios y computación en la nube exigen dominar estos conceptos.


Fin