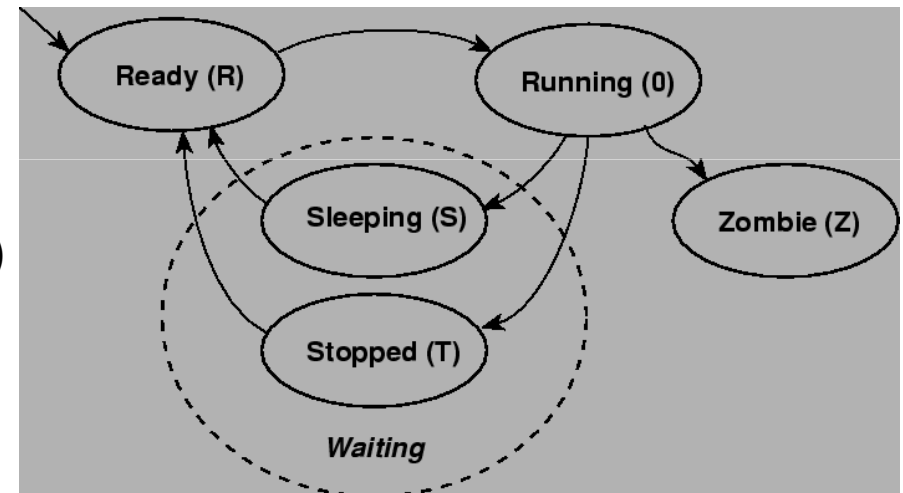


# Contenido

1. Introducción
2. Ejecución
  - 2.1 Comandos Asociados
3. Control de Procesos

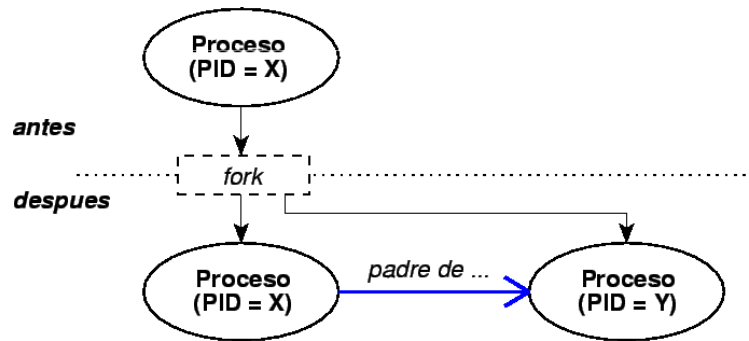
# 1. Introducción (I)

- **Proceso:** programa en ejecución.
  - No es únicamente código, ya que requiere asignación de memoria, uso de dispositivos de E/S, prioridad de ejecución.
- **Estados:**
  - Preparado (Ready, R): proceso listo para ejecutarse. En la cola de espera de la CPU.
  - En ejecución (Running, O): proceso en ejecución (CPU asignada).
  - Suspendido (Sleeping, S): proceso *preparado* que está en espera de algún evento (Ej. acabar E/S de datos) para poder seguir con su ejecución.
  - Parado (Stopped, T): sólo pasarán a *preparados* cuando reciban la señal que les permita continuar.
  - Zombie (Z): como veremos todo proceso tiene un padre. Al acabar, envía una señal a éste para que elimine su entrada de la tabla de procesos. Si por alguna razón el padre no recibe esta señal (por ej., ha finalizado antes que el hijo), el proceso no es eliminado de la tabla, quedando en estado “zombie”, que no consume CPU, pero sí otros recursos (memoria, por ej.) del sistema.



# 1. Introducción (II)

- **Identificación:** todo proceso en UNIX tiene un identificador (PID) único.
  - Es un número natural ( $\in [0, \text{Max}]$ )
  - La asignación se hace de manera consecutiva y circular.
- **Creación :**
  - En UNIX todo proceso se crea desde otro.
  - Veremos más adelante como hacer esto.
  - El proceso creado se denomina **hijo** y el proceso que lo creo **padre**.



- Cuando el sistema arranca, crea automáticamente el proceso **init** (PID=1). Este proceso es el responsable de crear todos los sucesivos procesos del sistema.
- Cuando un proceso termina, ejecuta una rutina especial, notifica al sistema y devuelve un código (número entero) de salida.
  - La variable “?” contiene el valor devuelto por el último proceso finalizado.

## 2. Ejecución

- Creación de un proceso desde línea de comandos:  
*Prompt> nombre\_fich\_ejecutable [argumento/s]*
  - A esta forma de ejecutar un proceso se le denomina en “primer plano” o “foreground”.
    - El shell permanece “inactivo” mientras se ejecuta el proceso. Sólo cuando acabe volverá el prompt del sistema y podremos seguir ejecutando comandos.
- Ejecución en **background**
  - El proceso se ejecuta en “**segundo plano**”, es decir, el proceso se va ejecutando mientras podemos seguir con nuestra sesión de trabajo.
  - Sintaxis: nombre\_fich\_ejecutable [argumento/s] **&**
  - El sistema muestra el PID asignado al nuevo proceso y a continuación el prompt.
  - El proceso shell asignado a la sesión será el padre del nuevo proceso.

## 2.1 Comandos

- **nohup**. Permite ejecutar comandos inmunes a señales de finalización o salida.
  - Se usa normalmente junto a la ejecución en background. En versiones anteriores de UNIX al eliminar al proceso padre, se eliminaban también sus hijos. **nohup** evita esto permitiendo dejar un proceso en ejecución una vez cerrada la sesión. Actualmente, al eliminar el padre UNIX asigna cualquier proceso hijo que estuviera en ejecución al proceso *init* (PID=1), de esta manera se evita la aparición de procesos *zombies*.
  - Sintaxis: **nohup** comando &
- **nice**. Permite cambiar la prioridad de un proceso (sólo a la baja...)
  - Sintaxis: **nice** [-numero] comando
- **at**. Permite programar la ejecución de un comando para una fecha y hora determinadas.
  - Debido a su peligrosidad, existe un control de uso. Normalmente suele prohibirse su ejecución a los usuarios normales.

## 3. Control de Procesos

- Comando **time**. Muestra el tiempo transcurrido durante la ejecución de un proceso, una vez finalizada ésta.
  - Sintaxis: **time** comando
  - Muestra el tiempo total de ejecución (**real**), el tiempo de CPU en modo usuario (**user**) y tiempo de CPU en modo sistema (**sys**).
- Comando **ps**. Muestra información acerca los procesos activos en un determinado momento.
  - Sintaxis: **ps** [opciones]
  - Opciones más importantes:
    - Por defecto muestra información en formato corto acerca de los procesos asociados a la sesión.
    - -e: muestra información de todos los procesos.
    - -f: listado completo: propietario, PID, PID del padre, tiempo de CPU, hora de comienzo, terminal asignado, tiempo de CPU acumulado y nombre del ejecutable.
    - -l: listado largo. Muestra entre otras cosas, el estado del proceso y su prioridad.
    - -u *nombre\_usuario*: muestra los procesos asociados a un determinado usuario.
- Comando **kill**. Manda **señales** a los procesos.
  - Señal: notificación asíncrona de eventos (stop, continue, terminate,...) a procesos.
  - Sintaxis básica: **kill** –señal PID/s
    - Finalizar un proceso: **kill -9 PID/s**