Procesos e Hilos

Un sistema informático debe ser capaz de dar soporte a la comunicación entre procesos y la creación de estos.

Un **proceso** (tarea), representa la ejecución de un programa individual. Su traza se refiere a la secuencia de instrucciones ejecutada por dicho proceso. Debemos distinguir lo que es un proceso de un programa, el proceso tiene un concepto dinámico, mientras que el programa tiene un concepto estático.

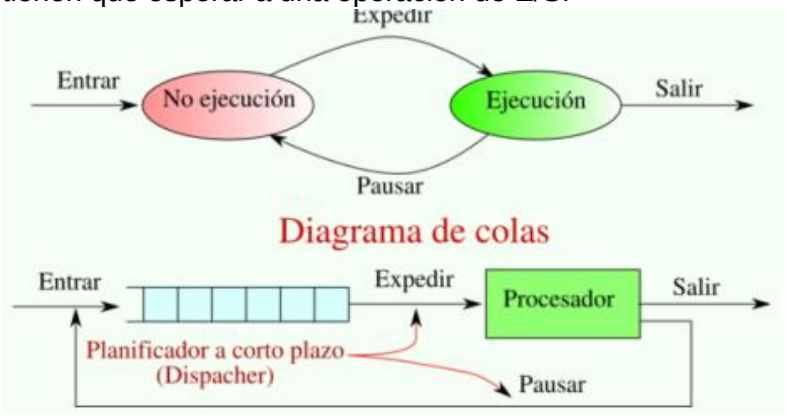
Los procesos están formados por:

- Un **BCP (Bloque de control de procesos)**, en el cual tenemos el PID, **estado**, prioridad, registros…
- Una **pila** con datos temporales, parámetros… Esta pila está dividida en:
  - Pila para el **núcleo**.
  - Pila para el **usuario**.
- **Datos** globales.
- **Texto** donde se encuentra el código.
- **Memoria compartida**, espacio de memoria reservada para la memoria compartida entre procesos.

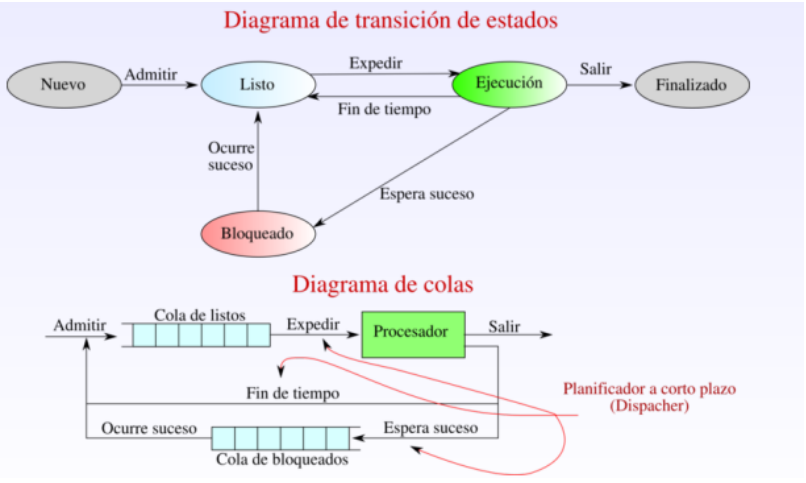
Estos procesos están guardados por el sistema. El sistema guarda punteros a BCPs de cada proceso.

Tenemos varios modelos de procesos dependiendo de los estados que estos tengan:

- **2 estados**: **Ejecución**, **no ejecución**. En el estado de no ejecución los estados se encuentran en una cola la cual no tiene prioridades, por lo que los estados salen de esta cola en orden. Esto es un problema ya que pueden haber procesos que necesiten salir antes o que no puedan salir ya que tienen que esperar a una operación de E/S.



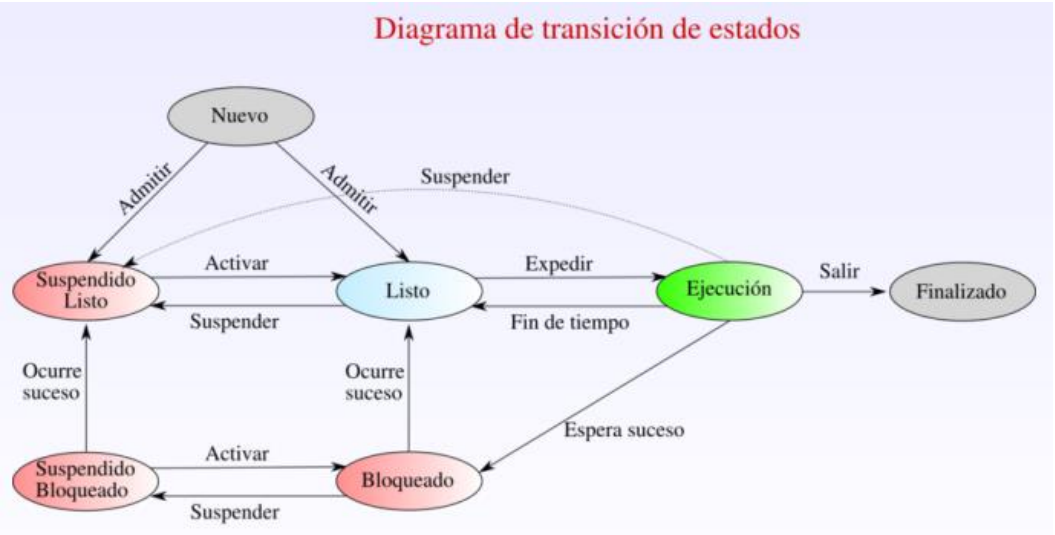
- **5 estados**: **Nuevo**, **listo**, **ejecución**, **bloqueado**, **finalizado**. En este modelo se añaden nuevos estados como el de listo, significa que los procesos con este estado están listos para ejecutarse en cuanto se les de la oportunidad, bloqueado, aquí están los procesos que no se pueden ejecutar debido a que esperan a que algún recurso de E/S esté libre, nuevo, procesos que se acaban de crear pero no están en memoria principal, y por último finalizado, procesos que han acabado y por tanto no se pueden volver a ejecutar.
  En este modelo los estados pueden cerrarse por varias razones, por terminación normal, por limite de tiempo excedido, por que no hay memoria disponible, se han violado los límites, errores de protección, errores aritméticos, tiempo máximo de espera rebasado, fallo de E/S, instrucción ilegal, instrucción privilegiada, mal uso de los datos, intervención del SO, solicitud del padre.



El problema de este modelo es que el procesador suele ser más rápido que E/S, por lo que es habitual que todos los procesos estén en memoria esperan por E/S y por lo tanto esta toda la memoria ocupada con procesos, pero ninguno se está ejecutando. La solución es pasar a disco procesos, así liberamos memoria.

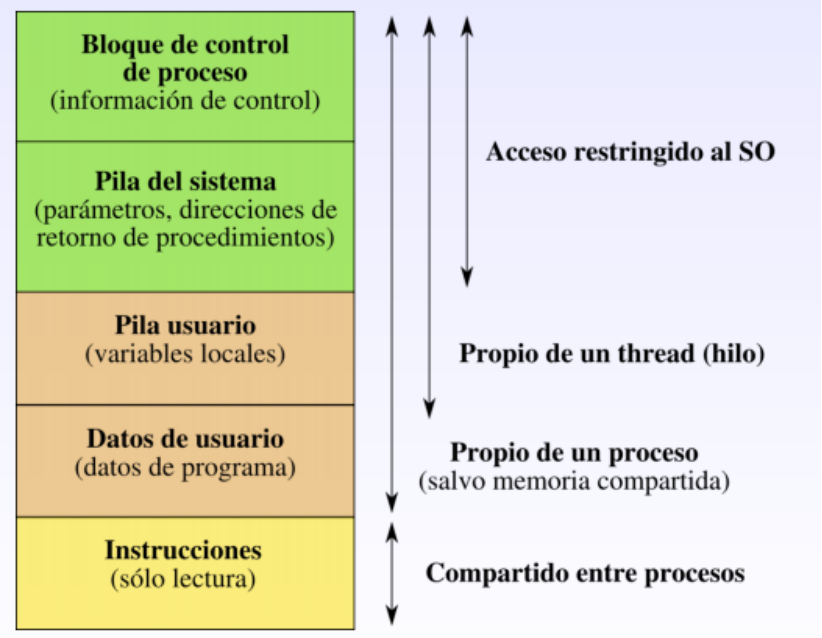
- **6 estados**: **Suspendido**. Se añade el estado suspendido donde el proceso o una parte de este se encuentra en el disco.
  En este modelo solo pueden suspenderse los estados que estén bloqueados, pero eso significa que los procesos en estado de listo siempre están ocupando memoria, por lo tanto es necesario poder suspender los procesos que estén en el estado de listo.

- **7 estados**: **Suspendido-listo, suspendido-bloqueado**. Cuando un proceso está en suspendido-bloqueado espera un evento para salir de ese estado., cuando un proceso está en suspendido-listo esta en el disco listo para ser ejecutado.



En el sistema operativo existen **estructuras de control** destinadas a mantener la información sobre el estado actual de cada proceso y de cada recurso. Para esto el sistema operativo construye **tablas** de información sobre cada entidad que esté administrando. Tenemos las siguientes tablas:

- Tablas de memoria: Almacenan información relacionada con la asignación de memoria, atributos de protección de bloques y cualquier información necesaria para gestionar la memoria virtual.
- Tablas de E/S: Contienen información sobre el estado y la operación del dispositivo E/S y la posición de memoria principal que se está utilizando como origen o destino de la transferencia de E/S.
- Tablas de archivos: Almacenan información de la existencia, posición, estado y otros atributos de los archivos.
- Tablas de procesos: Estás almacenan información sobre la ubicación del proceso, aquí se incluye: Instrucciones a ejecutar, datos para las variables locales y globales, constantes definidas y la pila. Se almacena también el BCP que incluye un conjunto de **metadatos** necesarios para la administración del proceso, los típicos son **ID**, **estado** y **ubicación**.



En la BCP se encuentra información como:

- Registros visibles al usuario: Son aquellos a los que pueden hacerse referencia por medio del lenguaje de máquina que ejecuta el procesador.
- Punteros de pila: Cada proceso tiene una o más pilas las cuales utiliza para almacenar parámetros, direcciones…
- Registros de control y de estado: Hay varios registros del procesador que se emplean para controlar su funcionamiento.
- Información de la planificación del estado: Estado, prioridad, suceso y la propia información de planificación que depende del algoritmo utilizado.

Un proceso se puede ejecutar de dos formas:

- Modo usuario. Menos privilegiado.
- Modo sistema, control o núcleo. Modo más privilegiado.

Los procesos se crean así:

1. Se asigna un **ID**.
2. Se asigna **espacio**.
3. Se inicia el **BCP**.
4. Se establecen los **enlaces** necesarios.
5. Se crean o amplían **estructuras de datos**.

Los procesos pueden cambiar de estado debido a:

1. Interrupción de reloj.
2. Interrupción de E/S.
3. Fallo de memoria.
4. Cepos (errores).
5. Llamada al sistema. Las llamadas corresponden a un procedimiento que lee los parámetros de la llamada y los pasa al SO.

¿Qué es lo que sucede exactamente cuando llega una interrupción? Se guarda el contexto del programa, se asigna al CP el valor de la dirección de comienzo del programa de atención a la interrupción. Cambia de modo usuario a modo núcleo para que se puedan ejecutar las instrucciones privilegiadas de la interrupción, y por último se ejecuta la rutina. Que se cambie el modo **no implica** que se cambie el proceso.

Si se quiere cambiar de proceso se debe, guardar el contexto del proceso 1, asignar el registro CP la dirección de comienzo de programa para la interrupción de cambio de proceso, cambiar a modo núcleo, ejecutar la rutina y, al ejecutar la rutina se actualiza la BCP, se mueve la BCP a la cola apropiada, se selecciona otro proceso, se actualiza ese proceso y se restaura el contexto del proceso.

Para crear procesos en **UNIX** se usa la función **fork()**. Esta función crea una nueva entrada en la tabla de procesos, asigna memoria y copia en ella los segmentos de datos y la pila del padre, y pone al hijo en estado de listo.

La función fork devuelve en el proceso hijo un 0, mientras que en el proceso padre devuelve el PID del hijo.

El hijo puede terminar de ejecutarse sin problema pero el padre debe, en algún momento, esperar a que el hijo acabe con la función **wait()**. Si el hijo acaba, pero el padre no ha hecho wait entonces el proceso hijo

es un **zombie**, ya que ha acabado pero el padre no hace el wait. Si el hijo acaba después del padre, el hijo se convierte en un **huérfano** mientras este ejecutándose, en el momento en el acabase su ejecución se convertiría en un zombie.

Otras funciones que crean procesos son: **execve**, esta función es usada para cargar un nuevo ejecutable binario en el espacio virtual de memoria del proceso. **vfork**, es igual que fork pero no copia los datos de la pila.

En UNIX tenemos identificadores como **uid**, que identifica al usuario que ha abierto la sesión, **gid**, identifica el grupo al que pertenece el usuario que inició la sesión, **euid**, identifica al usuario efectivo del fichero ejecutado, y **egid**, que identifica al grupo efectivo si se ejecuta un programa usando execve.

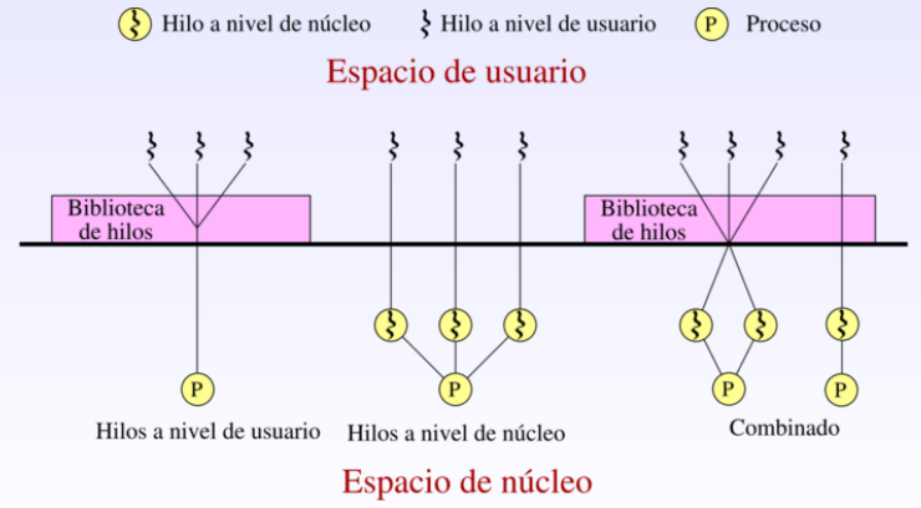
Para comunicar procesos se usan **tuberías (pipes)**, el proceso antes de hacer fork crea la tubería y después un proceso solo puede leer mientras que otro escribe.

También se pueden usar **señales** para la comunicación de procesos, estas señales pueden ser acumuladas o atendidas.

En los sistemas operativos no hay únicamente procesos también tenemos lo que se denominan **hilos**, estas son unidades de expedición. Los hilos tienen su propio bloque de control de hilo, poseen un estado de ejecución y un contexto, tiene una **pila de ejecución** que es usado para el almacenamiento de **variables locales**.

Los hilos pueden ser de **nivel de usuario** o **nivel de núcleo**.

- En los de nivel usuario, la aplicación biblioteca de hilos realiza todo el trabajo de gestión de hilos. El núcleo no sabe que esos hilos existen.
- En los de nivel núcleo, el núcleo mantiene la información de contexto del proceso y de los hilos. La planificación se realiza directamente en función de los hilos.



Los hilos a nivel de usuario tienen **ventajas** como que el intercambio no necesita privilegios del modo núcleo, se puede realizar planificación específica a nivel hilos, los hilos se pueden ejecutar en cualquier SO.

Las **desventajas** son que las llamadas al sistema suelen ser bloqueantes y no se aprovechan las ventajas de multiprocesadores.

Los hilos a nivel de núcleo tienen **ventajas** como que, si hay varios procesadores, es posible ejecutar en paralelo hilos del mismo proceso, y hay funciones multihilo. Las **desventajas** son que es necesario cambiar a modo núcleo para pasar de un hilo a otro.