

# UT1 - Introducción a la programación concurrente

---

Programación de servicios y procesos

# Objetivos:

---

En este primer tema vamos a conocer los conceptos básicos relacionados con la programación concurrente, así como la mayoría de la terminología que vamos a trabajar y utilizar durante todo el curso.

En un mundo en el que cada vez los dispositivos electrónicos son cada vez más potentes, y veloces, el software debe ser capaz de aprovechar las características que le ofrecen tanto el hardware como los sistemas operativos.

Son muchas las tareas que requieren de un procesamiento rápido de cantidades ingentes de datos. Un par de ejemplos los tenemos en las aplicaciones Big Data e Inteligencia artificial. Estos dos campos son unos de los máximos exponentes en cuanto a programación concurrente.

Los objetivos que alcanzaremos tras esta unidad son:

- Diferenciar entre programa y proceso
- Comprender qué es la concurrencia
- Conocer el concepto, diferencias y relación existente entre las dos unidades básicas de ejecución: procesos e hilos.
- Tener nociones sobre programación concurrente
- Entender el funcionamiento concurrente del SO y del hardware

► ¿Qué es para ti concurrencia?

# Contenidos:

---

- 1) **Introducción.**
- 2) **Programas. Ejecutables. Procesos. Servicios.**
- 3) **Procesos: Elementos de un proceso.**
- 4) **Estados de un proceso. Cambios de estado.**
- 5) **Hilos: Concepto y características.**
- 6) **Hilos vs. procesos.**
- 7) **Sistemas multitarea: Programación concurrente. Programación paralela. Programación distribuida.**
- 8) **Planificación de procesos por el sistema operativo.**

# Introducción.

---

- Los ordenadores son capaces de realizar muchas tareas de manera simultánea, pese a que el número de procesadores que tienen no es muy alto habitualmente. En parte es un engaño, ya que muchas veces las tareas se realizan “por turnos”, pero a tal velocidad que el ojo humano no es capaz de apreciar las discontinuidades en la ejecución.
- En otras ocasiones el procesamiento es verdaderamente simultáneo. Los procesadores modernos tienen varios núcleos de ejecución y cada uno de ellos funciona como un procesador de facto.
- También existe la posibilidad de utilizar varios ordenadores y construir una red en la que los elementos se distribuyan el trabajo para hacer bueno el dicho de que “la unión hace la fuerza”.
- Además del hardware, también el software influye en la realización de tareas de manera simultánea. Hoy todos los sistemas operativos son multitarea, pero no siempre ha sido así.

# Introducción.

El planificador de procesos es el elemento del sistema operativo que se encarga de repartir los recursos del sistema entre los procesos que los demandan. Determina la calidad del multiproceso del sistema y, como consecuencia, la eficiencia en el aprovechamiento de los recursos.

## Hardware

Los instrumentos

## Sistema Operativo

Director de Orquesta

## Software

Los músicos



# Programas, procesos y servicios.



## Programas

Una aplicación es un tipo de programa informático, diseñado como herramienta para resolver de manera automática un problema específico del usuario. Ejemplo: un navegador, Eclipse, Spotify, ...



## Procesos

Un proceso es un archivo que está en ejecución dentro del sistema operativo. Un proceso existe mientras esté ejecutando una aplicación, la cuál, puede implicar la generación de varios procesos.

## Servicios

Un servicio es un tipo de proceso que no muestra ninguna ventana ni gráfico en pantalla. Esto es debido a que están concebidos para que el usuario no los maneje directamente.



# Procesos: Elementos de un proceso.

---

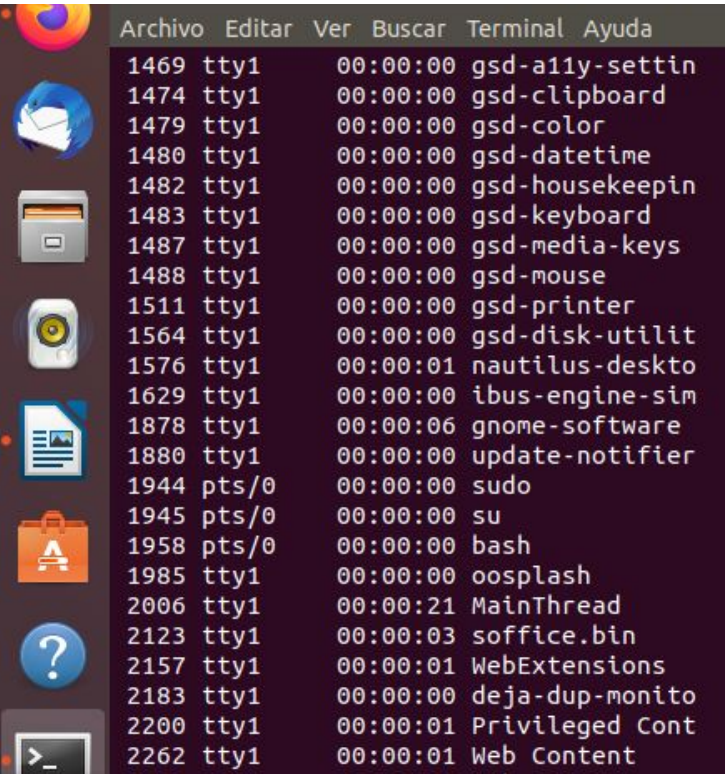
- De manera simplificada, se puede definir un proceso como un programa en ejecución (instancias de un navegador web, un procesador de texto, máquina virtual de java,...)
- Un proceso está compuesto por:
  - Las instrucciones que se van a ejecutar.
  - El estado del propio proceso.
  - El estado de ejecución, principalmente recogido en registros del procesador.
  - El estado de la memoria.
- Los procesos están entrando y saliendo constantemente del procesador. Se denomina **contexto** a toda la información que determina el estado de un proceso en un instante dado.
- Sacar un proceso del procesador para meter otro se conoce como **cambio de contexto**. Un cambio de contexto implica capturar el estado de la CPU y de sus registros, de la memoria y de la propia ejecución del proceso saliente para poder restaurar la información equivalente en el proceso entrante y poder continuar en el punto en el que este último abandonó el procesador en el cambio de contexto anterior.

# Procesos: Elementos de un proceso.

Los sistemas operativos tienen utilidades para consultar los Procesos en ejecución, junto con su PID (cada proceso contará con un PID, identificador de proceso, único) y datos relacionados con el consumo de recursos.

En un sistema Linux, utilizaremos el comando `ps` (**Process Status**) desde la terminal para ver los procesos en ejecución.

`ps -a` => Mostará todos los procesos en ejecución



	Archivo	Editar	Ver	Buscar	Terminal	Ayuda
1469	tty1		00:00:00	gsd-a11y-settin		
1474	tty1		00:00:00	gsd-clipboard		
1479	tty1		00:00:00	gsd-color		
1480	tty1		00:00:00	gsd-datetime		
1482	tty1		00:00:00	gsd-housekeepin		
1483	tty1		00:00:00	gsd-keyboard		
1487	tty1		00:00:00	gsd-media-keys		
1488	tty1		00:00:00	gsd-mouse		
1511	tty1		00:00:00	gsd-printer		
1564	tty1		00:00:00	gsd-disk-utilit		
1576	tty1		00:00:01	nautilus-deskto		
1629	tty1		00:00:00	ibus-engine-sim		
1878	tty1		00:00:06	gnome-software		
1880	tty1		00:00:00	update-notifier		
1944	pts/0		00:00:00	sudo		
1945	pts/0		00:00:00	su		
1958	pts/0		00:00:00	bash		
1985	tty1		00:00:00	oosplash		
2006	tty1		00:00:21	MainThread		
2123	tty1		00:00:03	soffice.bin		
2157	tty1		00:00:01	WebExtensions		
2183	tty1		00:00:00	deja-dup-monito		
2200	tty1		00:00:01	Privileged Cont		
2262	tty1		00:00:01	Web Content		



# Procesos: Elementos de un proceso.

---

En Windows usamos el comando tasklist desde la consola.

El procesador necesitará saber de cada proceso:

- PID, identificador de proceso, único.
- Estado actual
- Prioridad
- Ubicación en memoria

Para matar procesos:

- Linux: kill -9 PID
- Windows: taskkill /pid PID

También podemos ejecutar el administrador de tareas

```
C:\Users\jhorn>tasklist
```

Nombre de imagen	PID	Nombre de sesión	Núm. de ses	Uso de memor
=====	=====	=====	=====	=====
System Idle Process	0	Services	0	8 KB
System	4	Services	0	5.156 KB
Registry	124	Services	0	97.560 KB
smss.exe	704	Services	0	1.124 KB
csrss.exe	872	Services	0	6.360 KB
wininit.exe	984	Services	0	6.904 KB
services.exe	712	Services	0	10.820 KB

# Procesos: Elementos de un proceso.

Administrador de tareas							
Archivo Opciones Vista							
Procesos Rendimiento Historial de aplicaciones Inicio Usuarios Detalles Servicios							
^		4%	55%	0%	0%	2%	
Nombre	Estado	CPU	Memoria	Disco	Red	GPU	Motor de GPU
Aplicaciones (6)							
> Administrador de tareas		0,4%	30,4 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D
> Explorador de Windows		0,1%	50,1 MB	0 MB/s	0 Mbps	0%	
> Google Chrome (12)		0,1%	1.153,7 MB	0,1 MB/s	0 Mbps	0%	
> Herramienta Recortes		0,1%	2,4 MB	0 MB/s	0 Mbps	0%	
> Microsoft Edge (6)		0%	202,2 MB	0 MB/s	0 Mbps	0%	
> Procesador de comandos de Wi...		0%	0,8 MB	0 MB/s	0 Mbps	0%	
Procesos en segundo plano (...)							
> 64-bit Synaptics Pointing Enhan...		0%	0,2 MB	0 MB/s	0 Mbps	0%	
Aislamiento de gráficos de disp...		0,1%	4,5 MB	0 MB/s	0 Mbps	0%	

# ¿Qué es un proceso Daemon (Demonio)?

Proceso no interactivo que se ejecuta continuamente en segundo plano

**Son controlados por el Sistema Operativo y el usuario no puede intervenir**

**Proporcionan un servicio básico para el resto de procesos**

Como el de una impresora wi-fi cuando recibe una petición para imprimir, se activa, recibe el documento e imprime. Está siempre a la espera de ser activada para imprimir.

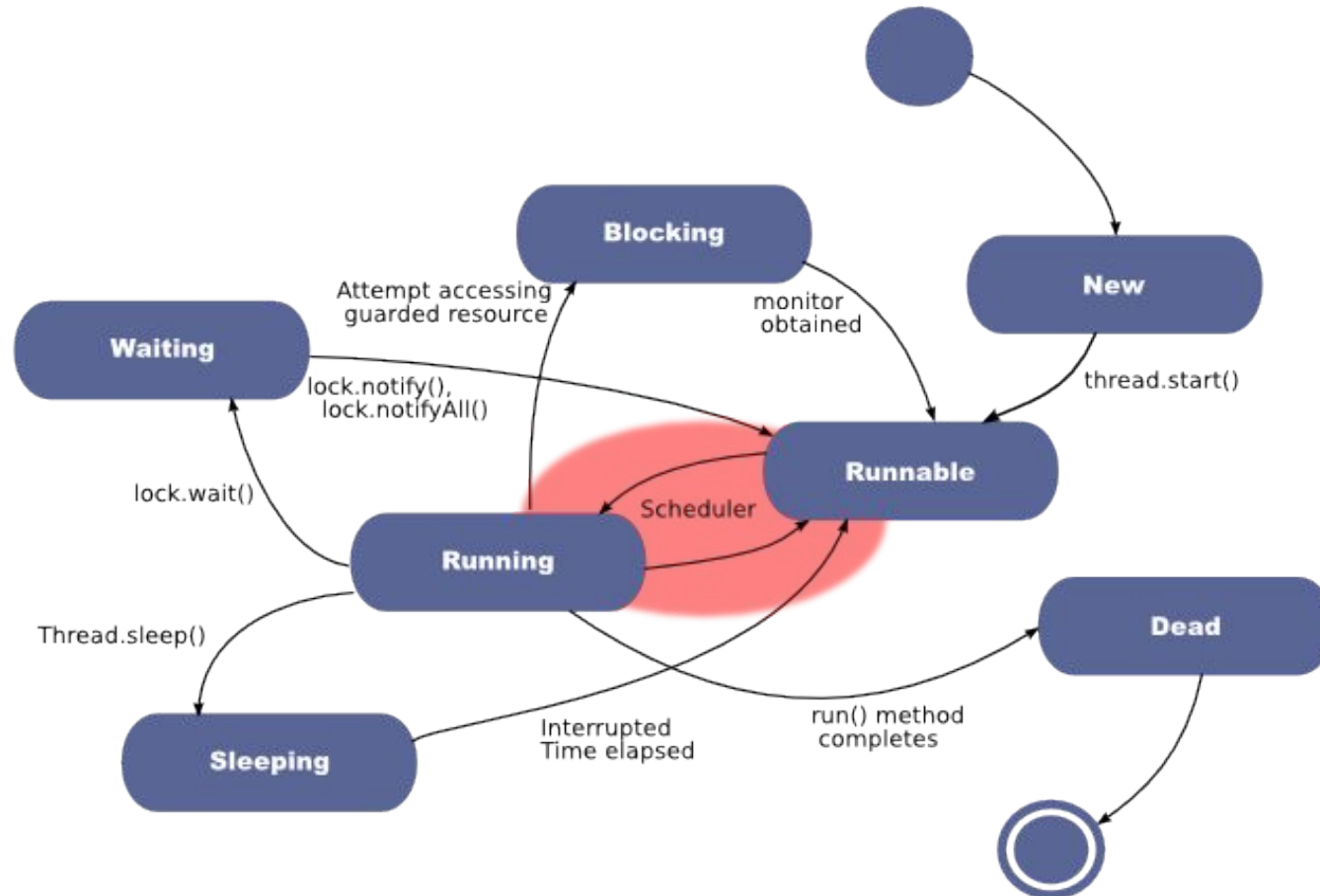
# Hilos: Concepto y características.

---

- Un **hilo** o **thread** es cada una de las tareas que puede realizar de forma simultánea una aplicación. Por defecto, toda aplicación dispone de un único hilo de ejecución, al que se conoce como hilo principal. Si dicha aplicación no despliega ningún otro hilo, sólo será capaz de ejecutar una tarea al mismo tiempo en ese hilo principal.
- Así, para cada tarea adicional que se quiera ejecutar en esa aplicación, se deberá lanzar un nuevo hilo o thread. Para ello, todos los lenguajes de programación, como Java, disponen de una API para crear y trabajar con ellos.

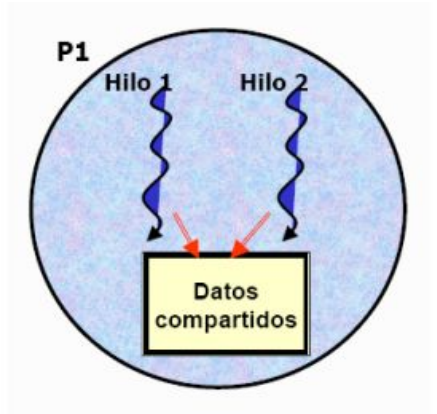


# Hilos: Estados de un hilo

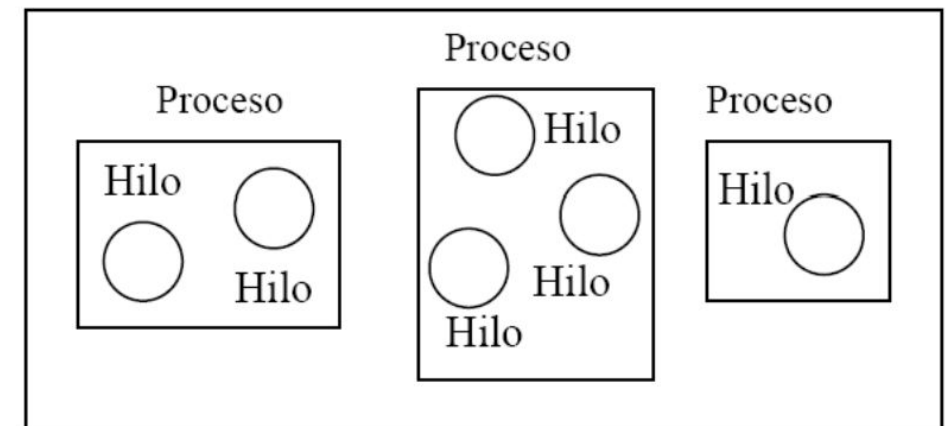


# Hilos vs. procesos.

- Un **hilo** es una secuencia de código en ejecución **dentro del contexto de un proceso**.
- Los **hilos no** pueden ejecutarse ellos **solos**.
- Requieren la supervisión de un proceso padre para correr.
- Dentro de cada proceso hay un hilo o varios hilos ejecutándose.
- La ventaja que proporcionan los hilos es la capacidad de tener más de un camino de ejecución en un mismo programa.



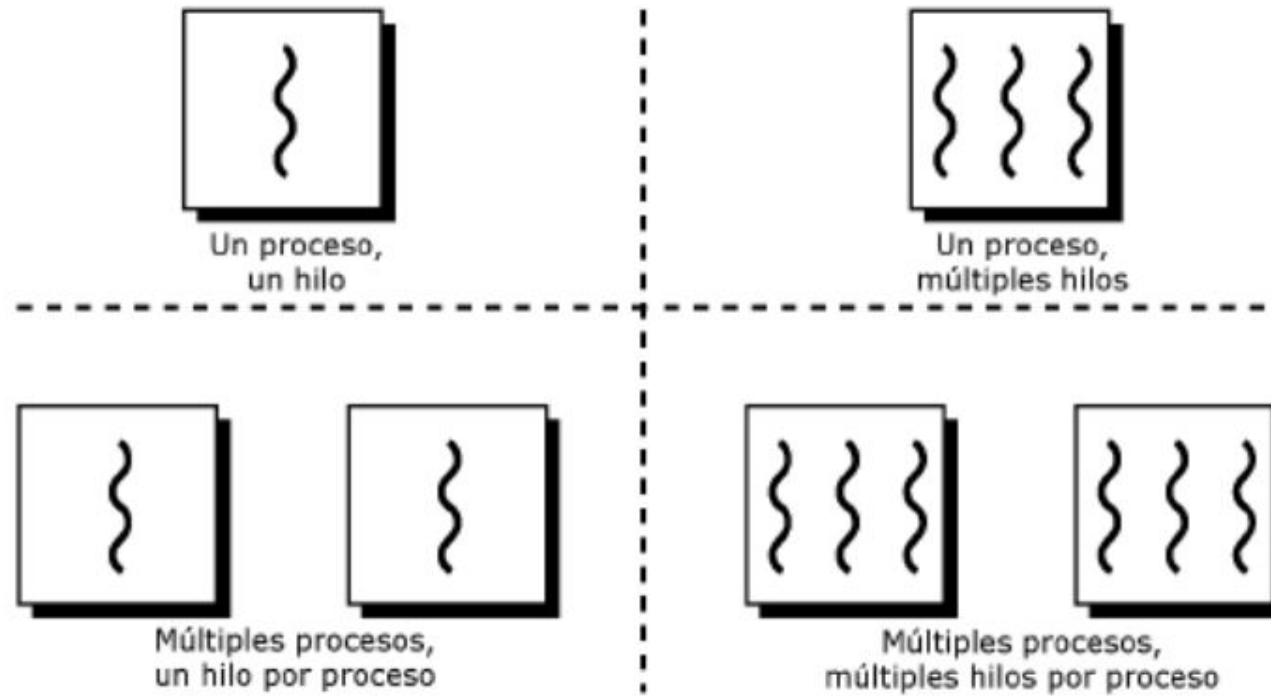
Word puede tener un hilo en background chequeando automáticamente la gramática de lo que se escribe, mientras otro hilo puede estar guardando automáticamente los cambios del documento en el que se trabaja.





# Hilos vs. procesos.

---



# Hilos vs. procesos.

---

Procesos	Hilos
Constan de uno o más hilos	Un hilo siempre existe dentro de un proceso
Son independientes unos de otros	Comparten los recursos del proceso de forma directa
Son gestionados por el SO	Son gestionados por el proceso
Se comunican a través del SO	La comunicación la controla el proceso

# Servicios

---



## Servicios

Un servicio es un proceso que, normalmente, es cargado durante el arranque del sistema operativo. Al no necesitar interacción con el usuario, los servicios suelen ejecutarse en forma de *\*demonios*, quedan su ejecución en *segundo plano*.

Recibe el nombre de servicio ya que es un proceso que queda a la espera de que otro le pida que realice una tarea. Como deben atender las solicitudes de varios procesos, los servicios suelen ser programas multihilo.

# Programación concurrente y paralela

---

Podemos decir que dos procesos son **concurrentes** cuando la primera instrucción de uno de los procesos se ejecuta después de la primera y antes de la última de otro proceso.

La planificación alternando los instantes de ejecución, **multitarea**, hace que los procesos se ejecuten de forma concurrente.

También la disponibilidad de varias unidades de proceso, **multiproceso**, permite la ejecución **simultánea o paralela** de procesos en el sistema. esta capacidad son las variantes de Unix, Windows NT y Mac OS X.

# Programa secuencial

---

Los programas secuenciales presentan una línea simple de control de flujo. Las operaciones de este tipo de programas están estrictamente ordenados como una secuencia temporal lineal.

El comportamiento del programa es solo función de la naturaleza de las operaciones individuales que constituye el programa y del orden en que se ejecutan (determinado por el conjunto de entradas que recibe).

En los programas secuenciales, el tiempo que tarda cada operación en ejecutarse no tiene consecuencias sobre el resultado.

# Programa concurrente

---

En los programas concurrentes existen múltiples líneas de control de flujo. Las sentencias que constituyen el programa no se ejecutan siguiendo un orden que corresponda a una secuencia temporal lineal.

En los programas concurrentes el concepto de secuencialidad entre sentencias continúa siendo muy importante. Sin embargo en los programas concurrentes es de orden parcial, mientras que, tal y como hemos comentado anteriormente, en los programas secuenciales era de orden estricto.

En los programas concurrentes la secuencialización entre procesos concurrentes se llama ***sincronización***.



## Reseña histórica

La naturaleza y los modelos de interacción entre procesos de un programa concurrente fueron estudiados y descritos por **Dijkstra** (1968), Brinch **Hansen** (1973) y **Hoare** (1974).

Estos trabajos constituyeron los principios en que se basaron los sistemas operativos multiproceso de la década de los 70 y 80.



# Comunicación y sincronización entre procesos

Cuando varios procesos se ejecutan concurrentemente puede haber procesos que colaboren para un determinado fin mientras que puede haber otros que compitan por los recursos del sistema.

En ambos casos se hace necesaria la introducción de mecanismos de comunicación y sincronización entre procesos.



## Programación concurrente

Precisamente del estudio de esos **mecanismos de sincronización y comunicación** trata la programación concurrente y este módulo de PSP.