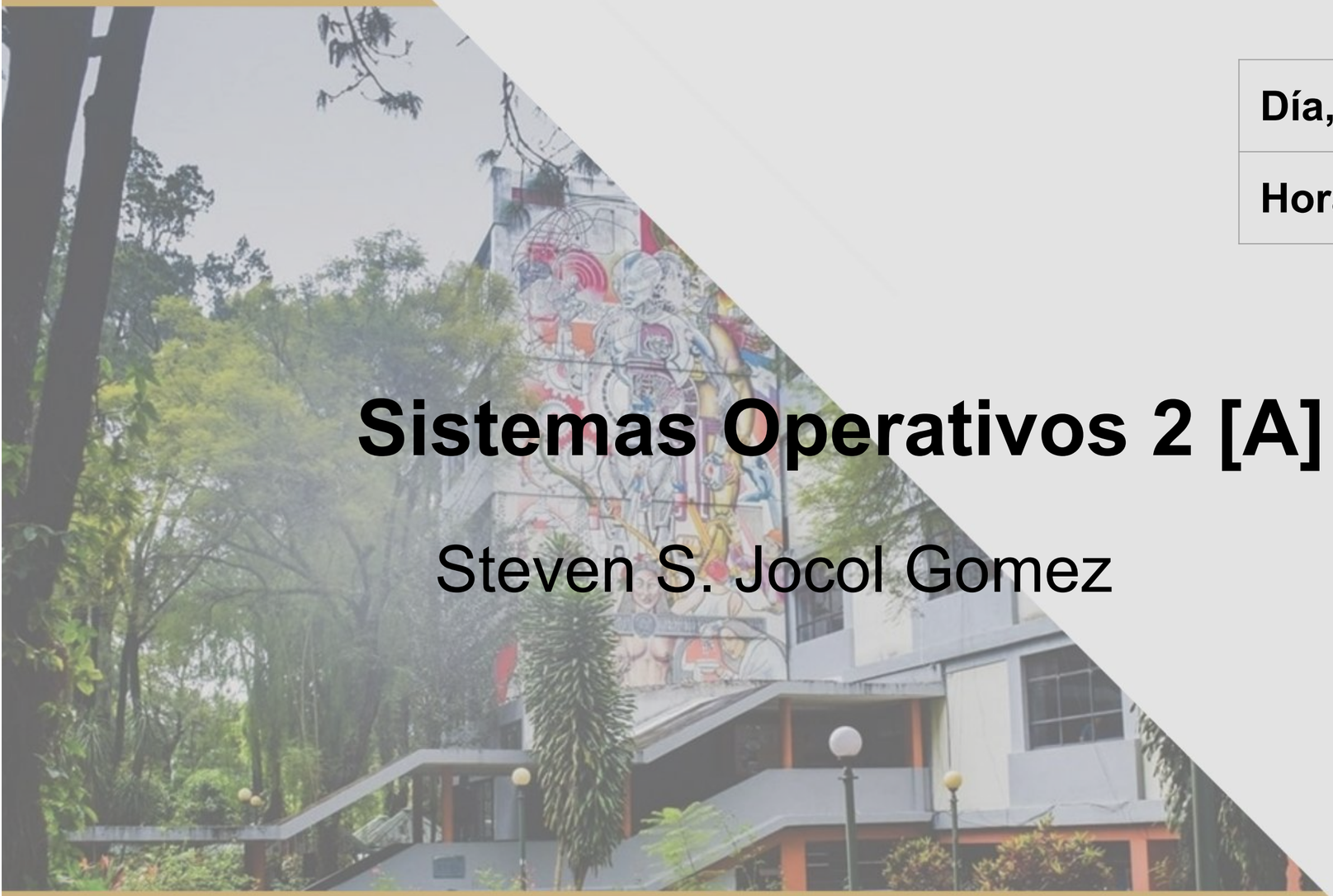
ESCUELA DE INGENIERÍA EN CIENCIAS Y SISTEMAS
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

ECYS | DTT
DESARROLLO DE TRANSFERENCIA TECNOLÓGICA



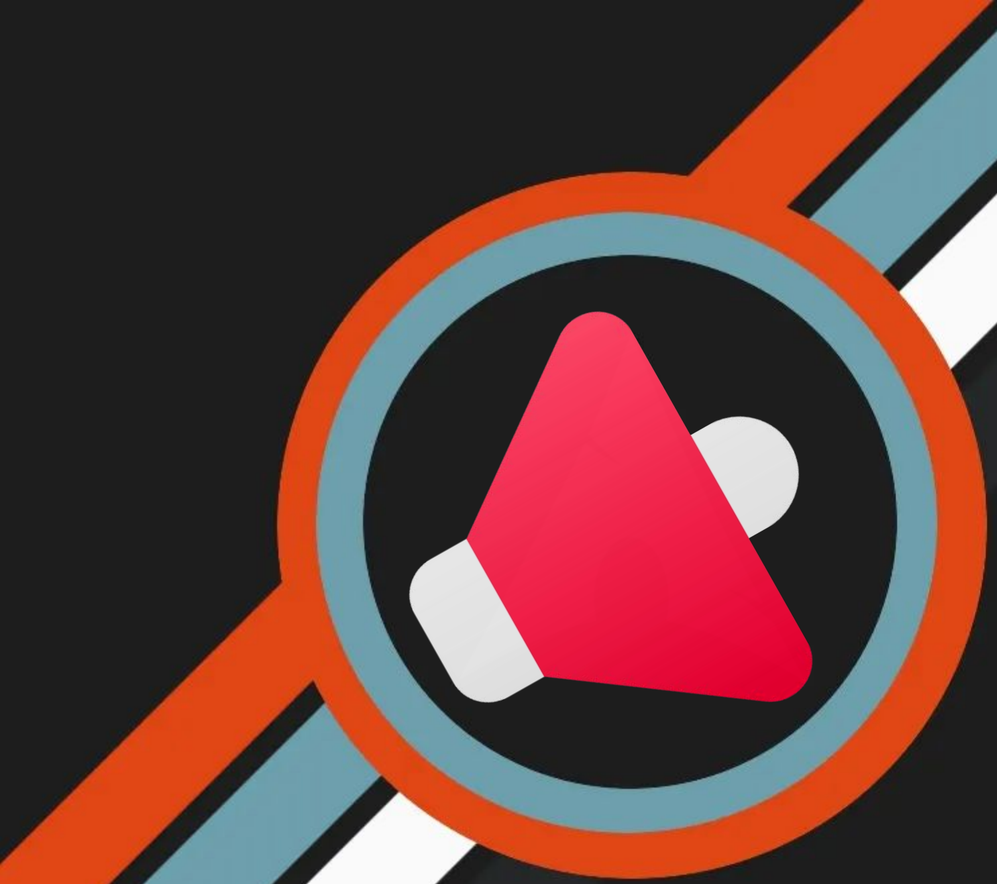
Día, Fecha:	Viernes, 08/08/2025
Hora de inicio:	17:20

Sistemas Operativos 2 [A]

Steven S. Jocol Gomez

1. UNIDAD 2:
PROCESOS E HILOS

Escuela de Ingeniería de Ciencias Y Sistemas
Facultad de Ingeniería
Universidad de San Carlos de Guatemala



Anuncios Importantes

Asignación DTT
Foro de la Semana
Entrega Hoja de Trabajo
Practica 1

AGENDA

1. Procesos y Gestión de Procesos
2. Estados de proceso y transiciones
3. Bloques de control de procesos (PCB)
4. Introducción a hilos

COMPETENCIA(S) QUE DESARROLLAREMOS

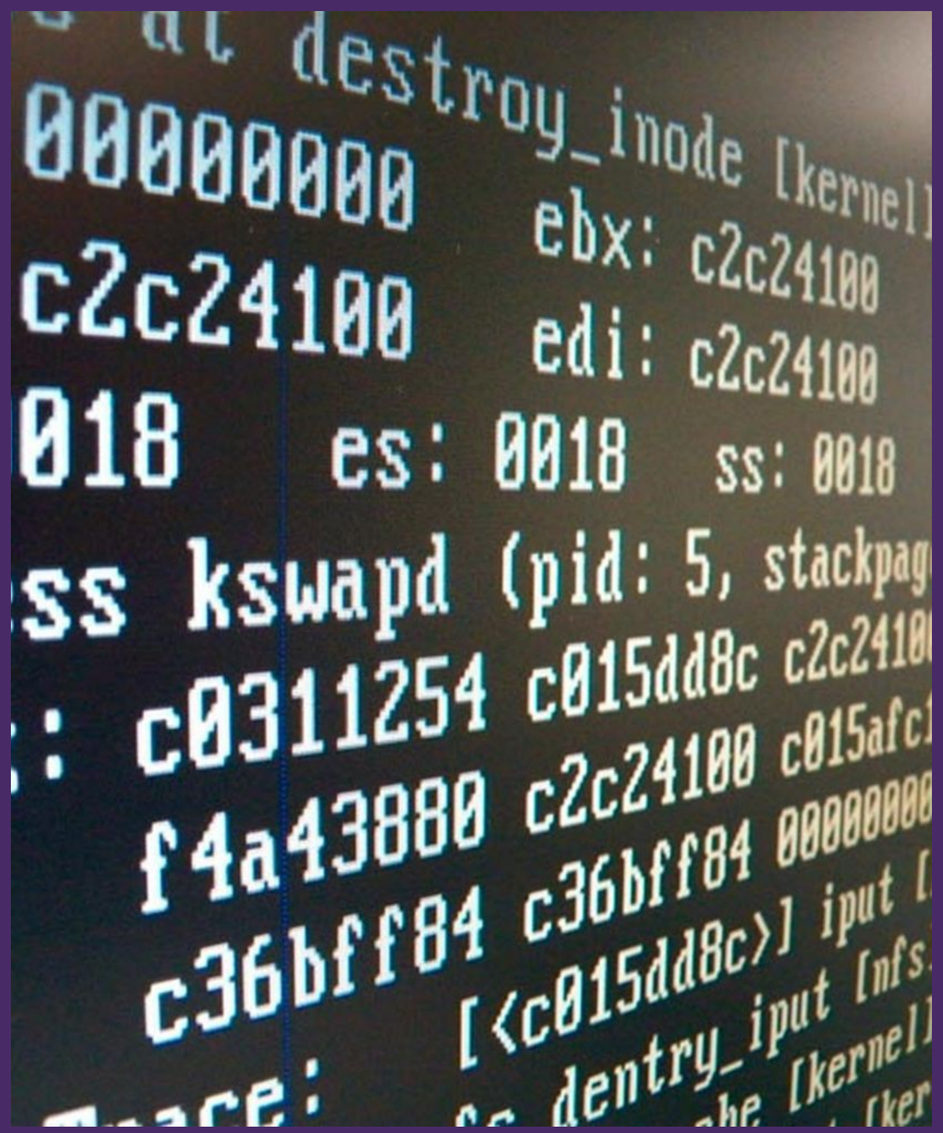
- Entender el marco de referencia o estructura lógica general de un sistema operativo, que le permita la utilización, análisis y diseño de sistemas operativos.
- Desarrollar e implementar nuevos sistemas operativos y modificar funcionalidades de sistemas operativos existentes

LABORATORIO SISTEMAS OPERATIVOS 2

GESTIÓN DE PROCESOS

¿QUÉ ES UN PROGRAMA?

video: https://www.youtube.com/watch?v=qgK0z08_Is4



CARACTERÍSTICAS DE UN PROCESO

Todo proceso sin importar el sistema operativo debe de tener las siguientes características fundamentales:

- ID de proceso, ID de grupo de proceso, ID de usuario e ID de grupo
- Ambiente
- Directorio de trabajo

Un proceso también proporciona un espacio de direcciones común y recursos comunes del sistema:

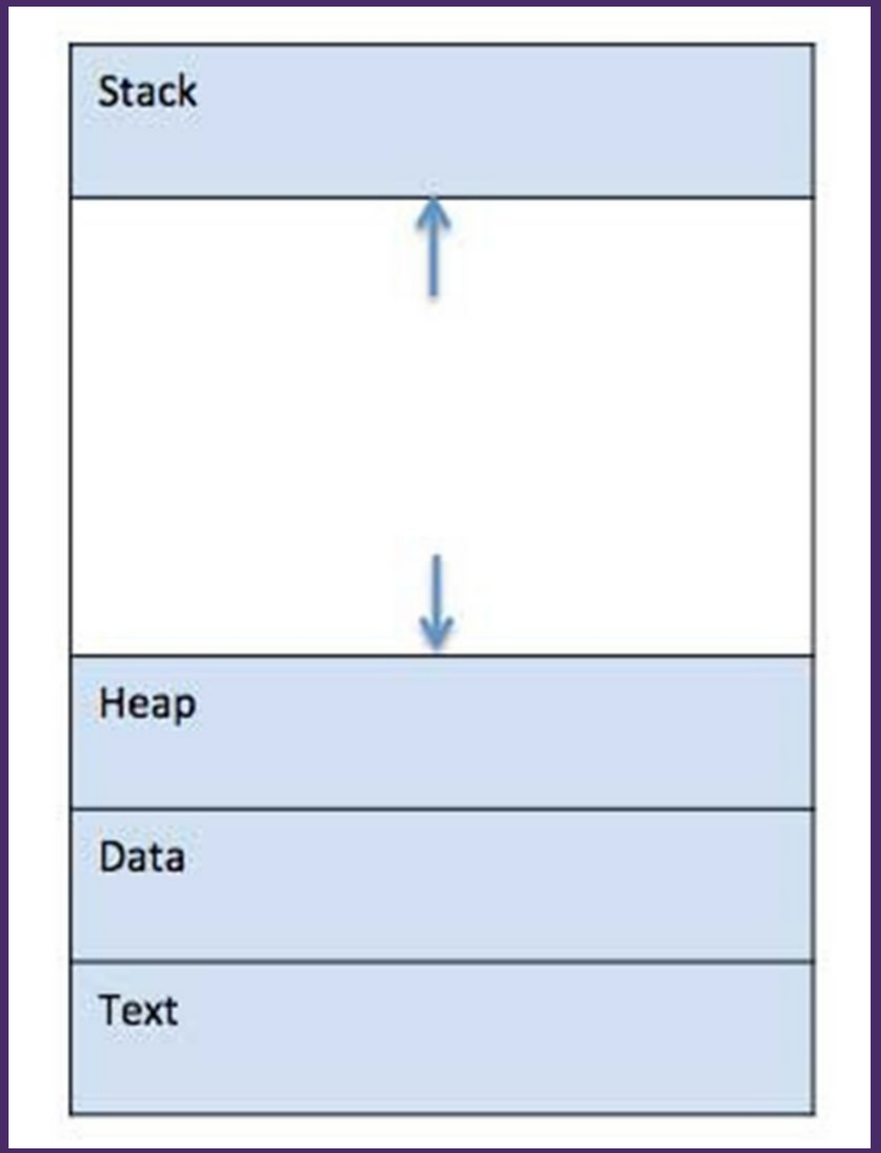
- Descriptores de archivos
- Acciones de señal
- Bibliotecas compartidas
- Herramientas de comunicación entre procesos (como colas de mensajes, pipes, semáforos o memoria compartida)

Cuando un programa se carga en la memoria y se convierte en un proceso, se puede dividir en cuatro secciones: pila (stack), montón (heap), texto y datos.

La "stack" es una estructura de datos que almacena información sobre las subrutinas activas de un programa de computadora y se utiliza como espacio temporal para el proceso.

Se distingue de la memoria asignada dinámicamente para el proceso que se conoce como "heap".

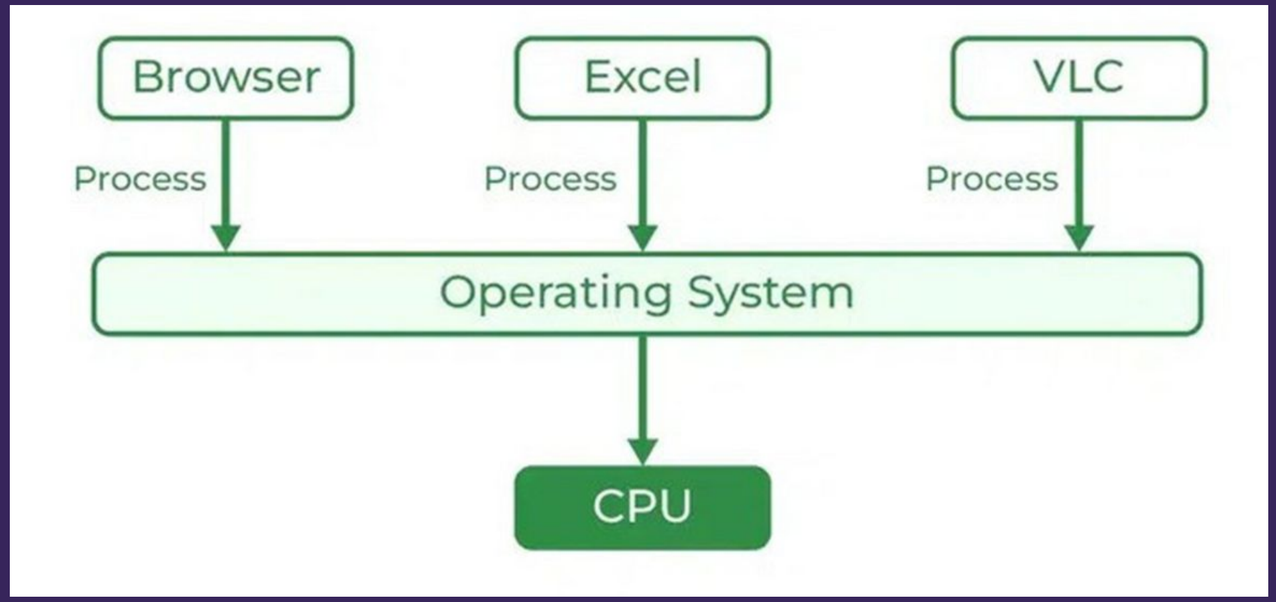
La seccion de texto o codigo incluye la actividad actual representada por el valor del contador de programa y el contenido de los registros del procesador, que pueden contener una instrucción, una dirección de almacenamiento u otro tipo de datos necesarios para el proceso.



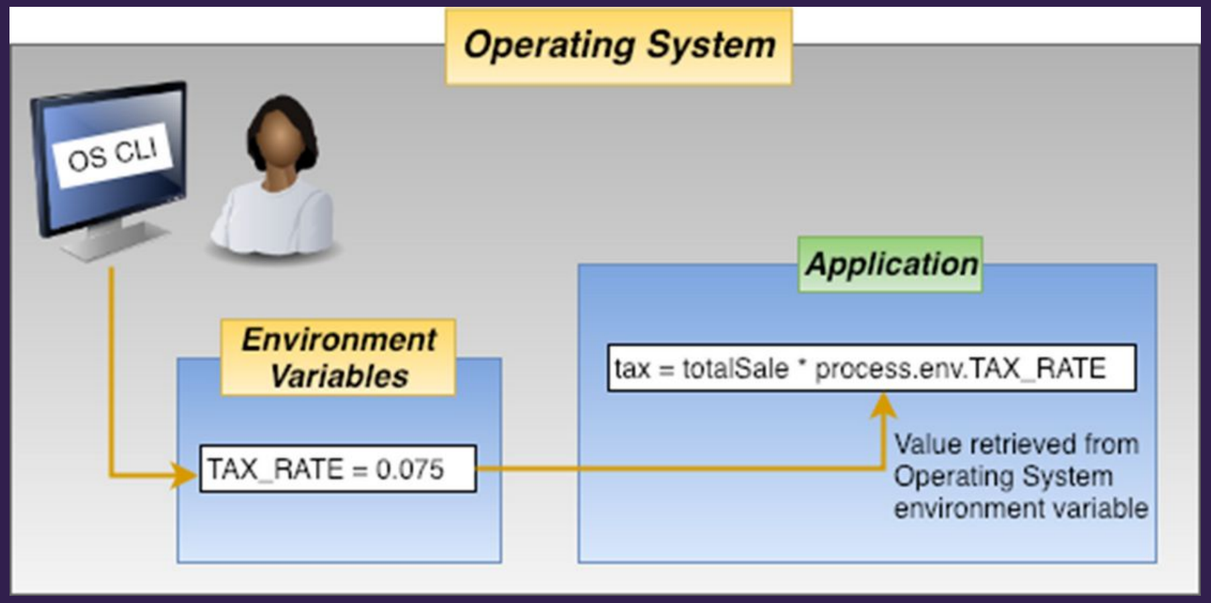
Puede haber varias instancias de un solo programa y cada instancia de ese programa en ejecución es un proceso.

Cada proceso tiene un espacio de direcciones de memoria independiente, lo que significa que un proceso se ejecuta de forma independiente y está aislado de otros procesos. No puede acceder directamente a datos compartidos en otros procesos.

Esta independencia de los procesos es valiosa porque el sistema operativo hace todo lo posible por aislar los procesos para que un problema con un proceso no corrompa ni cause estragos en otro proceso.1



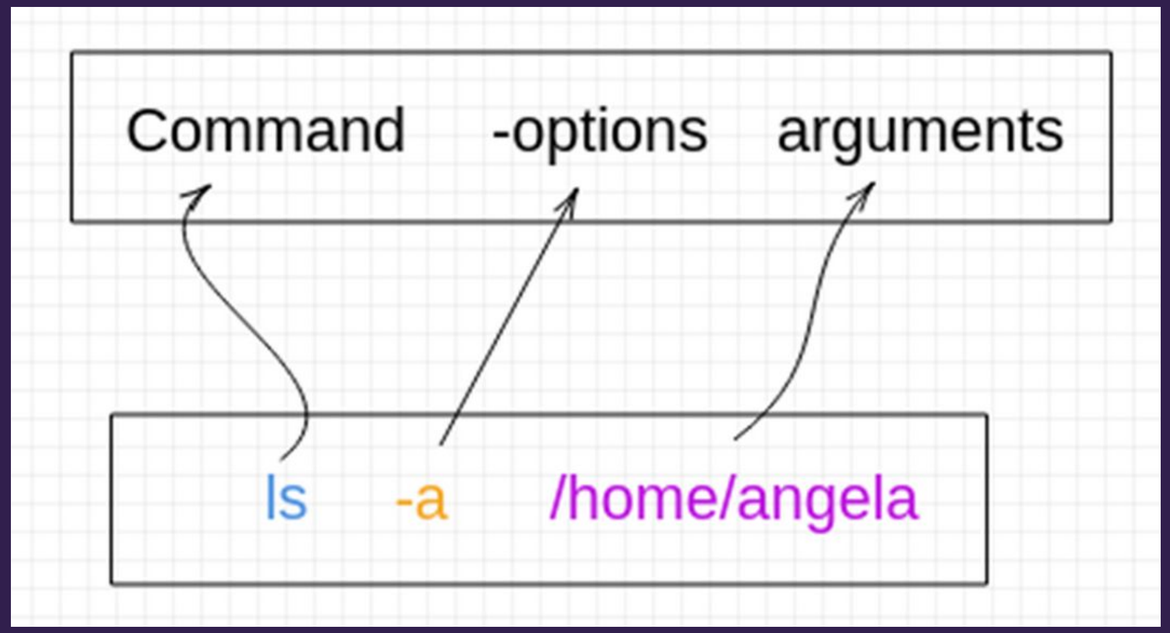
ENVIROMENT



El entorno o enviroment es una lista de variables de entorno heredadas del proceso principal que ejecuta una función exec.

Una variable de entorno contiene una cadena de caracteres de longitud variable. Por ejemplo, PATH es una variable del sistema exportada en /etc/environment que contiene una lista de directorios que utiliza el shell para buscar archivos ejecutables binarios al intentar ejecutar un programa.

ARGUMENTS



Los argumentos son una lista de valores de datos pasados a un nuevo proceso.

Los argumentos se ingresan en la terminal o consola después de ingresar el comando. Se puede escribir más de un argumento juntos, estos se procesarán en el ordenen que se escriban.

ESTADOS DE UN PROCESO

La vida de un proceso puede dividirse en un conjunto de estados, cada uno con ciertas características que describen su situación actual. Es fundamental comprender los siguientes estados:

- El proceso se está ejecutando actualmente en modo usuario.
  - Está realizando tareas normales en nombre de una aplicación o del usuario.

- El proceso se está ejecutando actualmente en modo kernel.
  - Está ejecutando código privilegiado del sistema operativo, por ejemplo, atendiendo una llamada al sistema.

- El proceso no se está ejecutando, pero está listo para ejecutarse tan pronto como el planificador lo seleccione.
  - Puede haber muchos procesos en este estado, y el algoritmo de planificación decide cuál se ejecutará a continuación.

- El proceso está durmiendo.
  - Un proceso se pone a dormir cuando no puede continuar ejecutándose, por ejemplo, cuando espera que una operación de E/S finalice.

ESTADOS DE UN PROCESO

- El proceso está listo para ejecutarse, pero el "swapper" (proceso 0) debe cargarlo en la memoria principal antes de que el kernel pueda programarlo para ejecutarse.

- El proceso está dormido, y el "swapper" lo ha intercambiado (swapped) a almacenamiento secundario para liberar espacio en la memoria principal para otros procesos.

- El proceso está regresando del modo kernel al modo usuario, pero el kernel lo interrumpe (preemption) y realiza un cambio de contexto (context switch) para programar otro proceso. La diferencia entre este estado y el estado 3 ("listo para ejecutarse") se explicará más adelante.

- El proceso ha sido recién creado y está en un estado de transición.
  - El proceso existe, pero aún no está listo para ejecutarse ni está dormido. Este es el estado inicial para todos los procesos excepto el proceso 0.

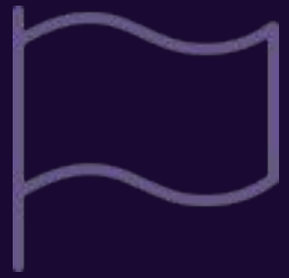
- El proceso ejecutó la llamada al sistema exit y está en estado zombi.
  - El proceso ya no existe, pero deja un registro con su código de salida y estadísticas de tiempo para que su proceso padre las recoja. El estado zombi es el estado final de un proceso.

NOTA

Dado que un procesador solo puede ejecutar un proceso a la vez, a lo sumo uno puede estar en los estados 1 (modo usuario) o 2 (modo kernel) simultáneamente. Estos dos estados corresponden a los dos modos de ejecución del procesador.

Estado	Descripción
Creado	El proceso ha sido creado recientemente por una llamada al sistema, pero aún no está listo para ejecutarse.
Ejecutándose en modo usuario	El proceso está en ejecución en modo usuario (proceso de usuario).
Ejecutándose en modo kernel	El proceso está ejecutando código del kernel (proceso privilegiado).
Zombi	El proceso ha terminado, pero permanece como zombi para que el padre recupere su código de salida.
Preemptado (interrumpido)	El proceso regresaba del kernel al modo usuario, pero fue interrumpido por el kernel.
Listo para ejecutar en memoria	El proceso está listo y en memoria, esperando ser programado por el kernel.
Listo para ejecutar, pero intercambiado (swapped)	El proceso está listo pero no se encuentra en memoria principal.
Dormido, intercambiado	El proceso está bloqueado y ha sido intercambiado al almacenamiento secundario.
Dormido en memoria	El proceso está bloqueado, pero reside en la memoria principal.

PCB: Process Control Block



```c
/* Simplified representation of the task_struct structure in Linux kernel */

struct task_struct {
    volatile long state;            // Process state (e.g., TASK_RUNNING, TASK_STOPPED)
    struct thread_info *thread_info;
    struct exec_domain *exec_domain; // Execution domain information (deprecated)
    struct mm_struct *mm;           // Memory management information (address space)
    struct fs_struct *fs;           // Filesystem information
    struct files_struct *files;     // File descriptor table
    struct signal_struct *signal;   // Signal handlers and signals pending
    struct sighand_struct *sighand; // Signal handling information
    ...
    /* Various other fields */
    ...
};
```

Un Bloque de Control de Procesos funciona como una representación del núcleo de un proceso , lo que permite al sistema operativo gestionarlo y controlarlo eficientemente. Conocido a menudo como PCB o estructura de tareas, es una estructura de datos que sirve como almacenamiento central de la información sobre un proceso.

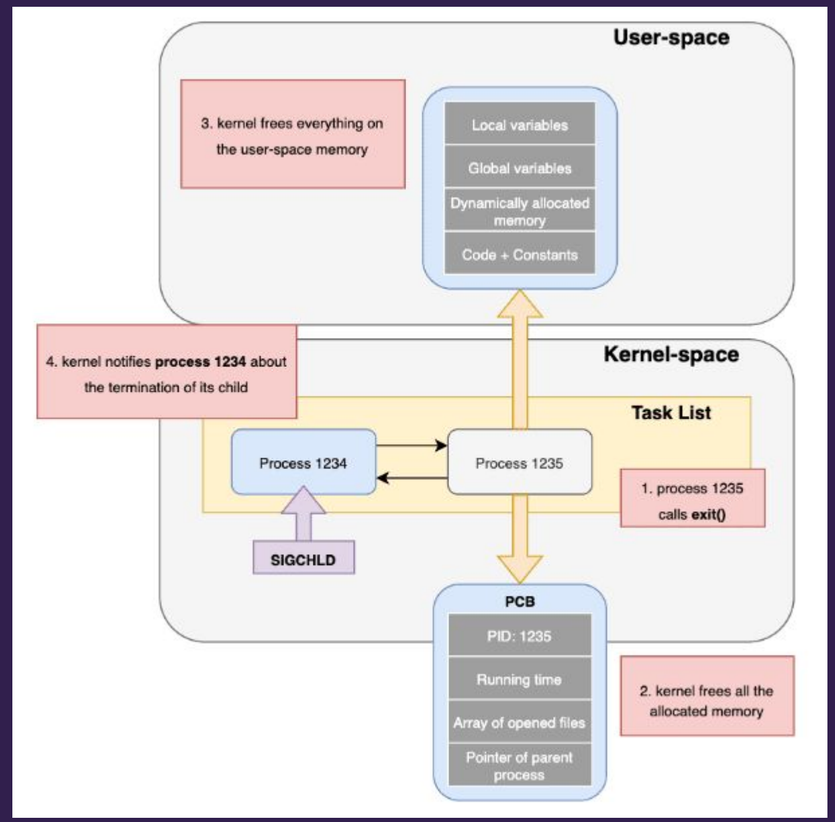
La creación de un nuevo proceso siempre acompaña la asignación de memoria para su task_struct, y esta estructura contiene toda la información relevante:

¿Dónde está la PCB?

Las PCB se definen principalmente dentro de la estructura task_struct .

Cada PCB es una instancia de esta estructura, cuyos campos almacenan información específica del proceso, como estado, registros y detalles de gestión de memoria.

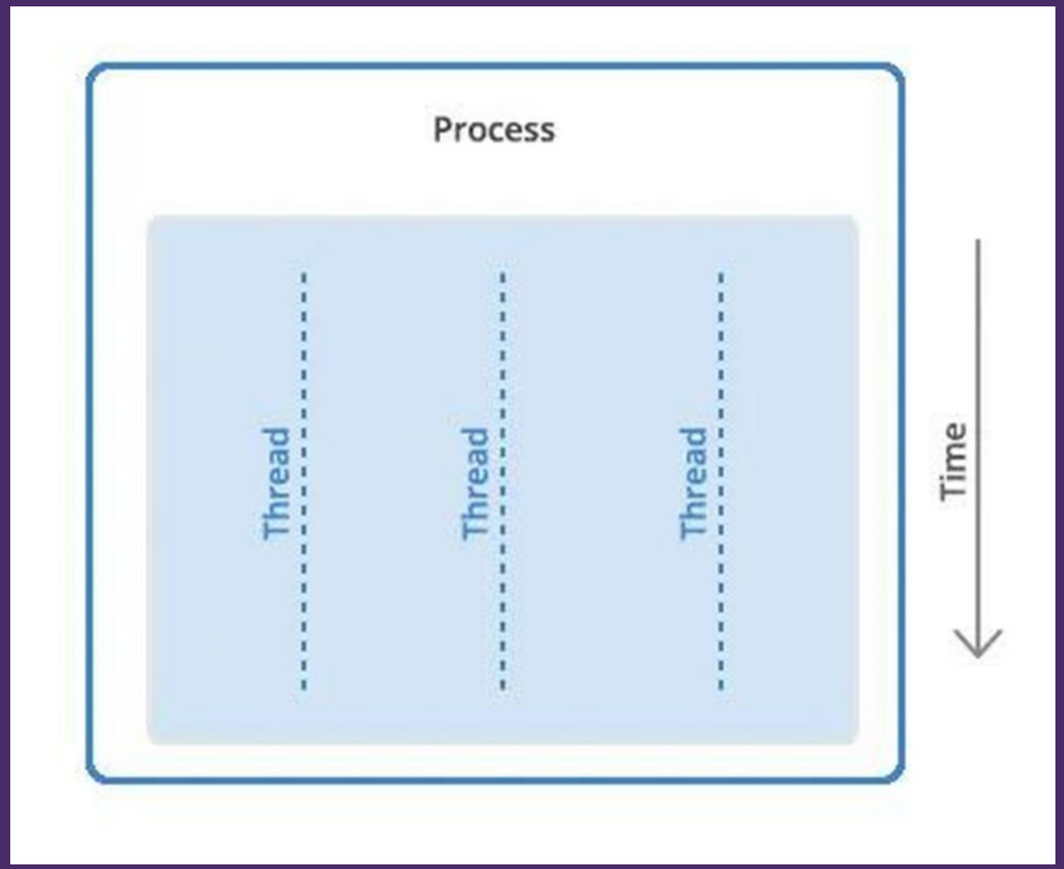
En Linux, la memoria tiene dos regiones centrales: el espacio de usuario y el espacio del kernel. Los procesos de usuario se ejecutan en un espacio de usuario asignado, separado del kernel. El espacio de memoria del kernel está reservado para el kernel de Linux y sus estructuras de datos (incluida la PCB).



LABORATORIO SISTEMAS OPERATIVOS 2

HILOS (INTRODUCCION)

HILOS (THREADS)

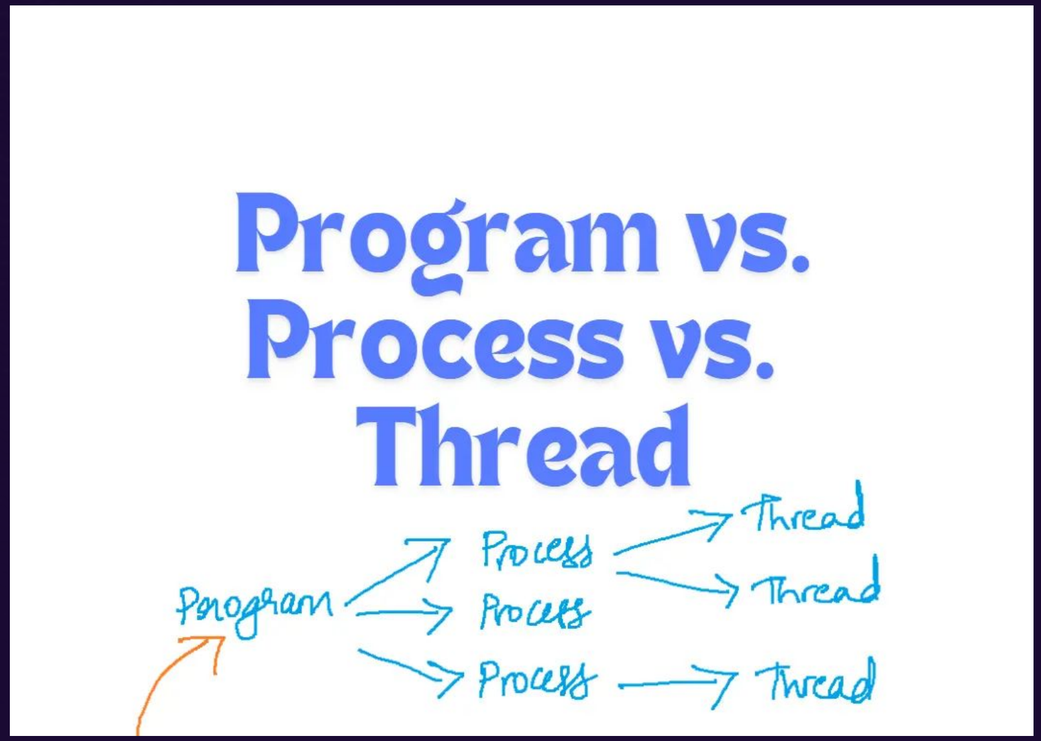


Un hilo de ejecución es la secuencia más pequeña de instrucciones que un programador puede gestionar de forma independiente.

El sistema operativo crea y administra hilos, estos comparten la misma memoria y recursos que el programa que los creó, permitiendo que múltiples threads colaboren y trabajen de manera eficiente dentro de un solo programa.

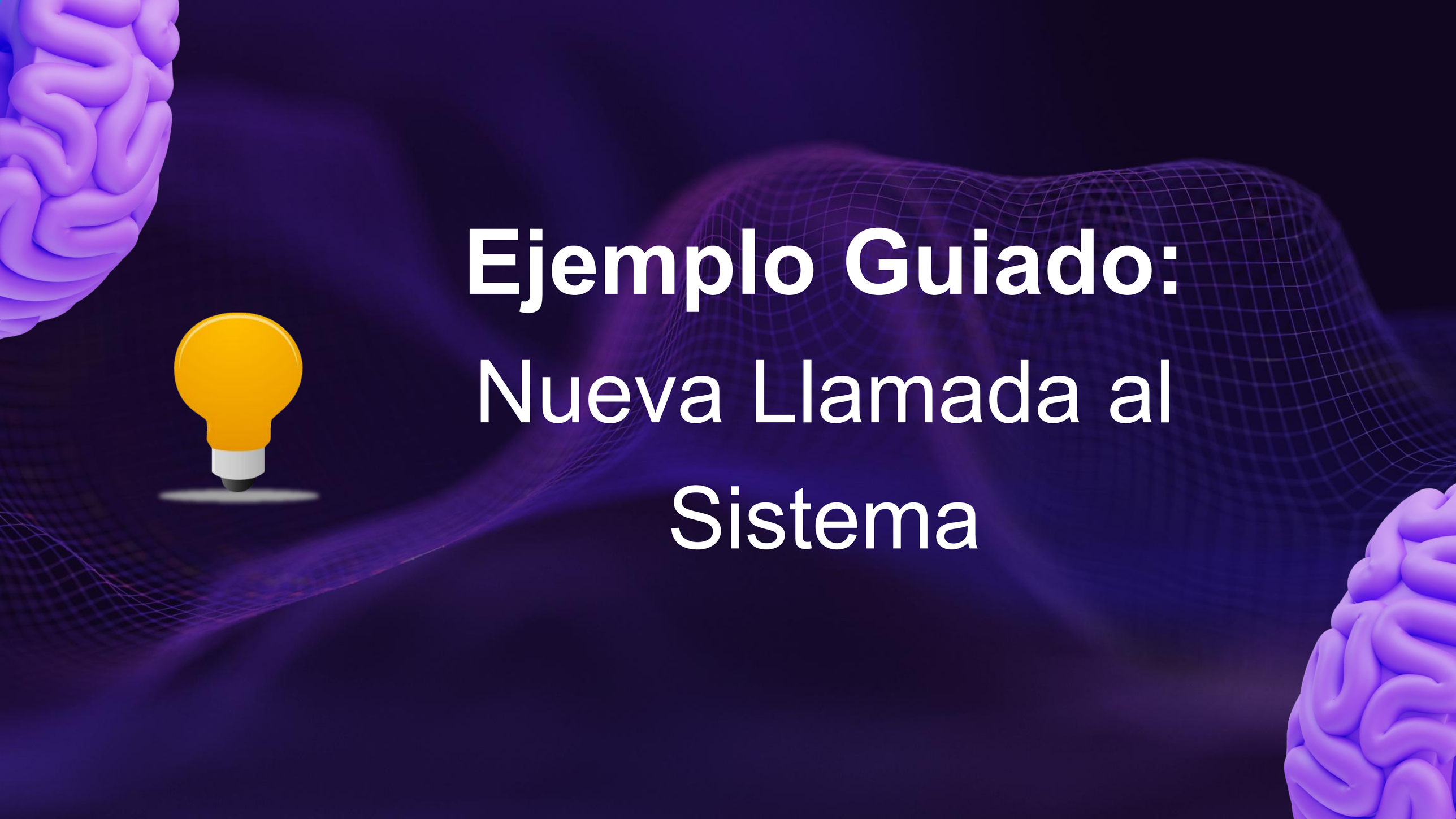
Proceso vs Hilo

En sistemas operativos modernos, tanto los procesos como los hilos son unidades de ejecución, pero se diferencian en el nivel de aislamiento, recursos que manejan y su costo de creación y cambio de contexto.



Puntos Clave

- Comparten los recursos del proceso: memoria, archivos abiertos, variables globales, etc.
- Se comunican fácilmente entre sí, pero también pueden interferirse si no se sincronizan.
- Crear un hilo requiere menos recursos del sistema que crear un proceso.
- El cambio de contexto entre hilos es más rápido.
- Existen hilos a nivel de usuario y a nivel de kernel
  - En sistemas modernos, la mayoría de los hilos son gestionados por el kernel (kernel-level threads), permitiendo un verdadero paralelismo.
  - Algunos lenguajes también manejan hilos de usuario (user-level threads) o green threads, más ligeros pero con menos integración con el os.


Ejemplo Guiado:
Nueva Llamada al Sistema

CONCEPTOS CLAVE APRENDIDOS

- **Características Fundamentales de un Proceso**
  Un proceso es un programa en ejecución que posee un espacio de direcciones independiente, identificadores únicos (PID, UID, GID), y se divide en secciones como pila, montón, código y datos, aislándose de otros procesos para garantizar estabilidad y seguridad.
- **Estados de Vida de un Proceso**
  Durante su ciclo de vida, un proceso transita por estados como nuevo, listo, en ejecución (modo usuario o kernel), en espera, detenido y zombi, gestionados por el planificador del sistema operativo para optimizar el uso de recursos.
- **El Bloque de Control de Procesos (PCB)**
  El PCB, o task_struct en Linux, es una estructura de datos que almacena toda la información necesaria para que el sistema operativo gestione un proceso, incluyendo su estado, registros, información de memoria y recursos asociados.

VALOR DE LA SEMANA

- **Consistencia:** Ser consistente en el aprendizaje es necesario para formar el cajón de herramientas con cada nuevo tema. Como estudiantes debemos demostrar consistencia en el aprendizaje y el curso para poder avanzar de manera adecuada.

REFERENCIAS

Process in Operating System
https://www.geeksforgeeks.org/process-in-operating-system/

Process States and Transitions
https://www.geeksforgeeks.org/process-states-and-transitions-in-a-unix-process/

The process control block
https://www.baeldung.com/linux/pcb

¡GRACIAS POR LA ATENCIÓN!

¿Dudas?