Ejercicio 1- Carga inicial de inventario

Resumen del ejercicio

Este programa de Sistemas Operativos simula un proceso de carga inicial de inventario utilizando un esquema de computación concurrente basado en procesos (fork). El objetivo es demostrar el uso de la comunicación entre procesos (IPC) y la sincronización para gestionar un recurso compartido.

Arquitectura del Sistema

El diseño se basa en una arquitectura clásica de productor-consumidor (o coordinador-trabajadores):

- **Proceso Coordinador (Padre):** Es el proceso principal que se encarga de la configuración del entorno (recursos IPC) y la supervisión del sistema.
- **Procesos Generadores (Hijos):** Actúan como productores, simulando la generación de datos de inventario. Estos procesos operan en paralelo y escriben la información de manera concurrente a un *buffer* compartido.

Características de los Datos de Salida

La salida del sistema es el archivo `salida.csv`, que documenta la carga de productos.

Campo	Descripción
ID	Identificador único del registro (secuencial, sin duplicados).
SKU	Código de producto único.
Cantidad	Stock inicial registrado.
Generador	Proceso hijo responsable de la creación del registro.

Debido a la naturaleza concurrente de la escritura, se espera que el orden de las filas en el archivo pueda variar con cada ejecución.

Guía de Operación y Pruebas

Compilación y Ejecución

Para iniciar el proyecto, el binario `tp1` se compila usando `make` en el directorio `ej1`.

El programa requiere especificar el número de generadores (`-g`) y la cantidad total de registros (`-n`).

Modo	Comando de Ejemplo	Propósito
Rápido (Producción)	`./tp1 -g 4 -n 5000`	Carga rápida para verificar la funcionalidad.
Lento (Demostración)	`./tp1 -g 5 -n 10000 -s 20000`	Ralentiza la producción (20,000 µs de espera) para observar la concurrencia y el manejo de IPC en tiempo real (ej. usando `htop` o monitoreando `/dev/shm`).

Validación y Monitoreo

La corrección de los datos generados puede validarse en cualquier momento con el script de AWK provisto: `awk -f scripts/validate.awk salida.csv`.

Para el monitoreo del sistema concurrente y los objetos IPC POSIX, se recomienda usar los siguientes comandos durante la ejecución en modo lento:

- `ps -o pid,ppid,comm -C tp1`: Para visualizar la jerarquía padre/hijo.
- `watch -n1 'ls -l /dev/shm'`: Para observar la vida útil de la Memoria Compartida y Semáforos.

Robustez y Terminación Controlada

Una parte crucial del proyecto es demostrar la robustez y la correcta limpieza de recursos (IPC) ante fallas:

Escenario de Falla	Acción de Prueba	Requisito del Sistema
Caída de Generador (Suave)	`kill -TERM <PID_hijo>` a un proceso generador.	El proceso coordinador debe manejar la interrupción y el sistema debe continuar hasta completar los N registros totales.
Terminación Ordenada	**Ctrl+C** o `kill -TERM <PID_padre>`.	El coordinador debe iniciar una rutina de *cleanup* (limpieza), asegurando que todos los hijos finalicen y que los objetos IPC en `/dev/shm` sean liberados.
Limpieza de Restos	`./scripts/cleanup_ipc.sh` después de una falla abrupta (`kill -KILL`).	Se verifica que los recursos persistentes hayan sido eliminados.

Autores

Grupo 04 Materia: Sistemas Operativos - 3649

Universidad Nacional de La Matanza

Segundo Cuatrimestre 2025 - Comisión 01-1900