Sistemas Operativos 2024/2025

Proyecto Final

Sebastián Jesús Rangel Urdaneta, Ángel Fernando Dal Nieto, 2262.

Pareja 6

1. Diagrama del sistema

El sistema implementado se basa en tres procesos principales: **Minero**, **Comprobador** y **Monitor**. La comunicación entre ellos se realiza mediante **memoria compartida**, **cola de mensajes** y **señales**.
El diseño responde a una arquitectura distribuida y sincronizada mediante semáforos anónimos.

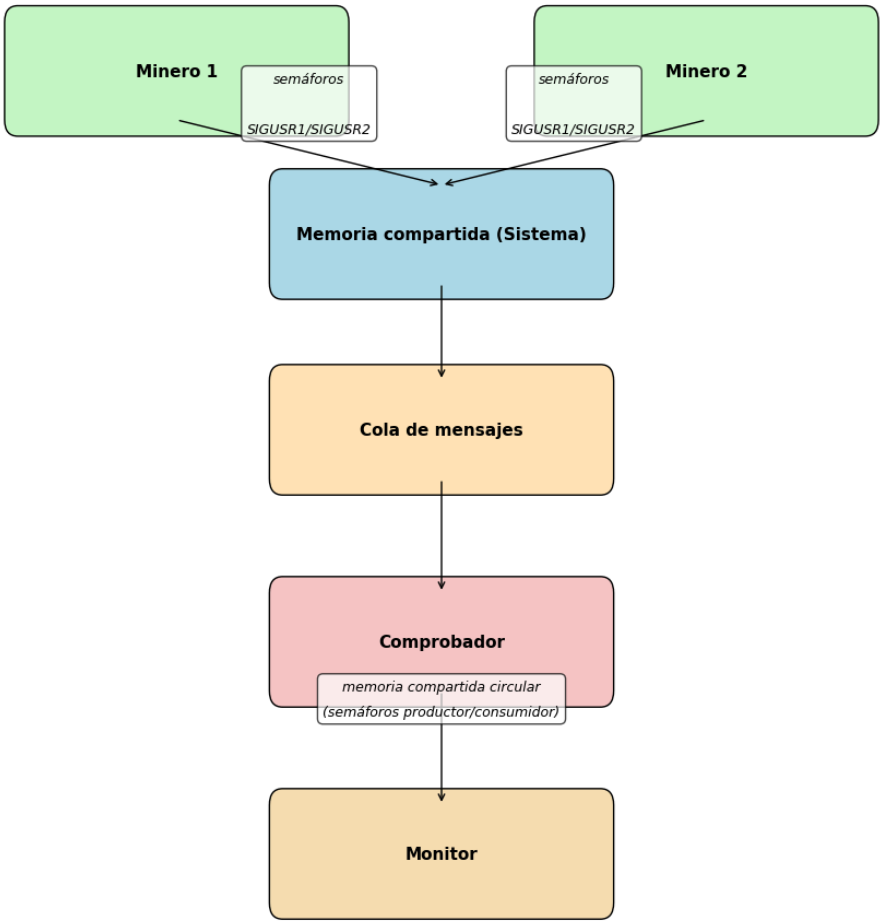
Componentes:

- **Proceso Monitor** (hijo del Comprobador): imprime bloques en pantalla
- **Proceso Comprobador**: recibe bloques por cola de mensajes y los valida
- **Proceso Minero**: se ejecuta en paralelo, con múltiples hilos, busca soluciones a la POW y gestiona rondas

Comunicación y sincronización:

- **Minero → Comprobador**: mediante **cola de mensajes POSIX** (**mq_send**, **mq_receive**)
- **Comprobador ↔ Monitor**: mediante **memoria compartida y semáforos anónimos**, siguiendo el esquema **productor–consumidor**
- **Mineros entre sí**: mediante **señales (SIGUSR1, SIGUSR2)**
- Todos los mineros acceden a una **memoria compartida común** que contiene el estado del sistema

No se ha implementado el proceso **Registrador** (opcional), por tanto, no hay comunicación por tuberías ni registros en fichero.



2. Descripción del diseño y problemas encontrados

Diseño implementado

- El sistema inicia con el proceso **monitor**, que crea e inicializa la memoria compartida del monitor, y lanza como hijo al proceso **monitor** que imprime los bloques.

- A continuación, se pueden lanzar múltiples procesos **minero**. Cada uno detecta si el sistema ya está inicializado:

    - El **primer minero** crea e inicializa la memoria compartida del sistema y los semáforos.

    - Los siguientes simplemente se conectan y esperan a que empiece una nueva ronda.

Rondas de minería

- Los mineros reciben una señal (**SIGUSR1**) para comenzar una ronda.

- Cada minero lanza varios hilos (según los parámetros del ejecutable) para buscar la solución de forma paralela.

- El primero en encontrar la solución intenta convertirse en **ganador** usando un semáforo exclusivo.

- Si lo logra, inicia una **fase de votación** enviando **SIGUSR2** al resto.

- Todos los mineros votan y, si se aprueba la solución, el bloque se envía a través de la cola de mensajes al comprobador.

Sincronización

- Se utilizan **semáforos anónimos POSIX** para:

    - Acceso seguro a memoria compartida (**mutex**)
    - Control de entrada/salida en buffer circular (**sem_fill**, **sem_empty**)
    - Coordinación de rondas (**mutex_ronda**, **ganador**)
- La entrada de nuevos mineros se gestiona mediante variables booleanas (**can_enter**) y semáforos adicionales (**entry_gate**, **entry_mutex**).

Problemas encontrados y soluciones

- **Coordinación entre señales y estados compartidos**: se soluciona introduciendo pequeñas esperas (**usleep**) y uso riguroso de semáforos antes de acceder a memoria.
- **Detectar al primer minero sin error**: se usó **shm_open** con **O_EXCL** para saber si la memoria ya existía.
- **Evitar que los nuevos mineros interrumpan la ronda en curso**: se bloquea la entrada durante la ronda mediante semáforos.

3. Limitaciones y pruebas realizadas

Limitaciones

- No se ha implementado el proceso **Registrador** (parte opcional del proyecto), por tanto:
    - No existe el hijo del proceso Minero que registre los bloques en fichero.
    - No se utiliza comunicación por **tuberías**.
- El sistema admite un número máximo de mineros (**MAX_MINERS**). Si se supera, no permite registrar más procesos.

Pruebas realizadas

- **Arranque secuencial** y **paralelo** de mineros con diferentes tiempos de vida y número de hilos.

- Se verificó la correcta creación de bloques, validación, votación y envío al monitor.

- Se observaron los bloqueos y desbloqueos del buffer circular de memoria compartida entre comprobador y monitor.

- Se comprobó que el **último minero** cierra correctamente todos los recursos y envía el bloque final al monitor.

- Se testearon interrupciones por señal (**SIGINT**) y por temporizador (**SIGALRM**) confirmando que el sistema libera recursos correctamente.