Juan Diego Cordero

C.I. 31.115.188

Solución Propuesta: La Cena de los Filósofos

La Cena de los Filósofos, propuesto por Edsger Dijkstra, es un problema clásico de sincronización de procesos en informática, diseñado para ilustrar los desafíos de la concurrencia y el riesgo del interbloqueo —deadlock— y la inanición —starvation—. El problema involucra *N* filósofos sentados alrededor de una mesa con *N* tenedores, uno entre cada dos filósofos. Además, se considera que cada filósofo requiere dos tenedores —el de su izquierda y el de su derecha— para comer.

En este sentido, se busca diseñar una solución óptima para prevenir el interbloqueo, maximizar el aprovechamiento de los recursos —tenedores— y evitar la inanición —que un filósofo muera de hambre—.

Planteamiento de la Solución

Una solución óptima, que garantiza la *imposibilidad lógica* del interbloqueo y evita la inanición —si se implementa correctamente el orden de asignación— es la introducción de una entidad central de control: *El Camarero*, para coordinar la limitación del número de filósofos que pueden estar en la *Sala de Espera* de los recursos.

1. **Diagrama de la Solución:** Imagine la mesa de los filósofos, pero con un *Camarero* de pie en el centro.
    a. ***Filósofos* —Procesos—:** Los *N* filósofos.
    b. ***Tenedores* —Recursos—:** Los *N* tenedores compartibles.
    c. ***El Camarero* —Monitor/Semáforo Central—:** Una entidad que actúa como árbitro, limitando el acceso al recurso compartido.

2. **Mecanismo del Algoritmo:** Un filósofo, antes de intentar acceder a cualquier tenedor, debe solicitar permiso al *Camarero* para ingresar a la sala para comer.
    a. **Límite de Concurrencia:** El *Camarero* mantiene un contador que limita el número de filósofos que pueden estar en la mesa a *N-1*. Para *N=5* filósofos, el límite es *4*.
    b. **Arbitraje:** Si hay menos de *N-1* filósofos activos —en la mesa—, el *Camarero* permite la entrada. Si el límite ya se alcanzó —*N-1* filósofos activos—, el filósofo entrante espera hasta que otro filósofo termine de comer y libere su cupo.
    c. **Adquisición de Tenedores:** Una vez que el filósofo obtiene el permiso —ha ingresado a la sala de espera—, procede a intentar tomar el tenedor izquierdo y luego el derecho. La clave es que, dado que solo hay —como máximo— *N-1* filósofos intentando comer, al menos uno siempre podrá tomar ambos tenedores.

d. **Liberación de Recursos:** Después de comer, el filósofo libera sus dos tenedores y notifica al *Camarero* que ha terminado, liberando así un cupo en el límite de concurrencia.

Criterio de Optimalidad	Solución del Camarero
Imposibilidad Lógica del Interbloqueo	**Garantizada.** Originalmente, el interbloqueo ocurre porque todos los filósofos toman el recurso izquierdo y esperan indefinidamente por el derecho. Al limitar el número de filósofos que pueden intentar comer simultáneamente a *N-1*, siempre queda un tenedor libre, asegurando que al menos un filósofo pueda adquirir los dos recursos que necesita.
Ausencia de Inanición	**Puede ser evitada.** Si el *Camarero* implementa una política de asignación equitativa —por ejemplo, una cola *First In, First Out*—, se garantiza que ningún filósofo espere indefinidamente.
Máxima Concurrencia	**Óptima.** En el caso de *N=5*, el límite de *N-1* permite que, como máximo, dos filósofos coman a la vez, maximizando el aprovechamiento de los recursos dada la disposición de los tenedores. Esta solución permite la máxima concurrencia posible sin riesgo de bloqueo.

Analogía con un Sistema Operativo (SO)

El problema de *La Cena de los Filósofos* es una abstracción perfecta de la gestión de recursos compartidos por procesos concurrentes en un Sistema Operativo:

Elementos del Problema	Elemento en el Sistema Operativo (SO)
Filósofos	Procesos —o hilos— que requieren el acceso a recursos.
Tenedores	Recursos de E/S —por ejemplo, archivos, bases de datos, impresoras, buffers de memoria—.
Comer/Pensar	Sección crítica —uso del recurso— / Procesamiento no crítico.
Interbloqueo	Dos o más procesos —o hilos— están permanentemente bloqueados, esperando cada uno por un recurso que está siendo retenido por otro proceso de ese mismo conjunto.
Camarero	Un *Monitor/Semáforo Central de Conteo* que controla el número total de procesos que pueden acceder simultáneamente a un conjunto de recursos.