Informe Técnico de Concurrencia: Implementación del Sistema SIGET

1. Introducción y Propósito del Sistema

El Sistema de Gestión Empresarial y Tecnología (SIGET) se ha diseñado como una infraestructura crítica distribuida para la administración masiva de servicios públicos. En el despliegue de este entorno, la **concurrencia** no es opcional, sino una capacidad primordial para gestionar múltiples procesos e hilos en un mismo periodo temporal.

La arquitectura de SIGET busca mitigar el denominado **'cuello de botella de Von Neumann'**. Para superar las limitaciones del procesamiento secuencial, el sistema implementa técnicas de **multiprogramación** —cargando diversos programas en memoria— y **procesamiento multinúcleo**, aprovechando las unidades de procesamiento físico para ejecutar tareas de forma independiente. Mediante una intercalación controlada y un **cambio de contexto** (context switch) eficiente, el sistema garantiza que la CPU permanezca ocupada al 100 %, optimizando el rendimiento global en escenarios de alta demanda interactiva.

2. Objetivos Primordiales del Diseño

El diseño orientado a la concurrencia en SIGET persigue dos objetivos técnicos fundamentales basados en el análisis de riesgos del sistema SSSM:

- **Garantía de integridad:** Se debe asegurar la **exclusión mutua** para evitar errores lógicos graves. Por ejemplo, en la gestión de inventario, si dos hilos intentan reservar simultáneamente el **"último módem disponible"**, el sistema debe impedir una doble asignación. Solo un hilo adquiere el derecho de acceso mientras el otro espera, manteniendo la consistencia total de los datos.
- **Prevención del colapso:** Ante la concurrencia de miles de usuarios y agentes, el sistema debe ser responsivo. La gestión robusta de los hilos permite que SIGET sea fluido y seguro, alternando el uso de recursos entre procesos activos sin degradar la experiencia del usuario final.

3. Arquitectura del Problema Productor-Consumidor en SIGET

La interacción entre los módulos de SIGET se fundamenta en el modelo **Productor-Consumidor**, permitiendo una comunicación asincrónica y un desacoplamiento efectivo:

- **Productor:** El módulo de gestión de pedidos actúa como tal, generando eventos ante nuevas órdenes o modificaciones de servicio.
- **Consumidor:** El módulo de inventario procesa estos eventos para asignar recursos físicos (cableado, equipos).

Para la mediación, se emplean **colas de mensajes** que gestionan los datos siguiendo un orden **FIFO (First-In-First-Out)** o basado en prioridades, asegurando el **aislamiento entre procesos**. Este mecanismo optimiza el flujo: si la cola está vacía, el consumidor se bloquea; si está llena, el productor espera. Este diseño garantiza que un módulo no sobrecargue al otro, permitiendo una alta escalabilidad en la arquitectura orientada a servicios.

4. Mecanismos de Sincronización y Control de Acceso

Como Arquitecto de Sistemas, he determinado el uso de semáforos y primitivas de C++ para proteger las secciones críticas y regular el ciclo de vida de los procesos:

- **Semáforos Binarios (Mutex):** Implementados mediante std::mutex o std::lock_guard, garantizan que solo un hilo acceda a recursos sensibles, como el saldo de un cliente en el módulo de facturación, asegurando que las actualizaciones sean estrictamente atómicas.
- **Semáforos Contadores:** Utilizados para gestionar el acceso a pools de recursos limitados.

- **Operaciones Atómicas de Dijkstra:**
    - **wait(S):** Decrementa el valor del semáforo. Si el resultado es **menor que cero**, el proceso se bloquea y entra en estado de espera.
    - **signal(S):** Incrementa el valor del semáforo. Si existen procesos bloqueados, el sistema se encarga de **despertar** a uno de ellos para que reanude su ejecución activa.

5. Análisis de Seguridad: Condiciones de Carrera e Interbloqueos

La estabilidad de SIGET exige la neutralización de errores difíciles de depurar mediante un diseño **thread-safe**.

Condiciones de Carrera Se evitan imponiendo exclusión mutua estricta. Sin este control, la reserva de equipos o el procesamiento de pagos dependería del orden aleatorio de los hilos, comprometiendo la fiabilidad transaccional.

Prevención de Interbloqueos (Deadlocks) Para evitar el bloqueo permanente, SIGET impone un **orden estricto en la adquisición de recursos**. Si el Hilo A y el Hilo B requieren acceso al **Recurso X (registro de cliente)** y al **Recurso Y (registro de inventario)**, ambos deben solicitarlos siempre en la misma secuencia jerárquica (primero X, luego Y), rompiendo así la posibilidad de una espera circular.

Condición	Definición Técnica
Mutua exclusión	Al menos un recurso está asignado a un solo proceso a la vez de forma no compartida.
Retención y espera	Un proceso está reteniendo recursos ya asignados mientras solicita recursos adicionales.
No expropiación	Los recursos no pueden ser retirados forzosamente; deben ser liberados voluntariamente.
Espera circular	Existe una cadena de procesos donde cada uno espera un recurso retenido por el siguiente.

6. Conclusiones sobre la Fiabilidad del Sistema Distribuido

La robustez de SIGET reside en su capacidad para transformar la teoría de sistemas operativos en una solución de ingeniería de alta densidad. La correcta implementación del IPC (Inter-Process Communication) mediante **Pipes (tuberías)** para comunicaciones unidireccionales locales y **Sockets/RPC** para la interacción entre módulos distribuidos garantiza la tolerancia a fallos. Al maximizar la eficiencia de la CPU y aplicar un control de sincronización avanzado, SIGET se consolida como una infraestructura capaz de mantener la continuidad del servicio y la integridad absoluta de los procesos concurrentes.