

## **Caso de Estudio - Sistema de Gestión de Emergencias Médicas**

### **Contexto**

El Departamento de Salud ha solicitado a la IUDigital de Antioquia desarrollar un sistema de gestión para el Centro de Emergencias Médicas de la ciudad. Este sistema debe coordinar la asignación de recursos (ambulancias, personal médico, equipamiento) a incidentes de emergencia que ocurren con frecuencia variable y en diferentes lugares de la ciudad.

El sistema debe gestionar múltiples llamadas de emergencia simultáneas, priorizar según gravedad, asignar recursos limitados, monitorear el estado en tiempo real, y coordinar la comunicación entre diferentes equipos de emergencia.

### **Desafío**

Los estudiantes deben diseñar e implementar un sistema que simule la gestión concurrente de emergencias médicas, enfrentando los siguientes desafíos:

1. Procesamiento de llamadas concurrentes: Las llamadas de emergencia pueden llegar en cualquier momento y deben ser atendidas sin bloquear el sistema.
2. Gestión de recursos compartidos: El sistema cuenta con recursos limitados (ambulancias, médicos, equipamiento) que deben ser asignados de manera óptima evitando condiciones de carrera.
3. Priorización de emergencias: No todas las emergencias tienen la misma gravedad. El sistema debe implementar un algoritmo de priorización que considere:
  - Gravedad del incidente (crítico, grave, moderado, leve)
  - Tiempo de espera
  - Distancia geográfica
4. Monitoreo en tiempo real: El sistema debe actualizar constantemente el estado de cada emergencia y de los recursos disponibles sin afectar el rendimiento.
5. Comunicación entre equipos: Los equipos de emergencia (ambulancias, hospitales, centros de despacho) deben comunicarse entre sí para coordinar la atención.
6. Tolerancia a fallos: El sistema debe manejar situaciones como la caída de conexiones o recursos que se vuelven temporalmente inaccesibles.

### **Requerimientos técnicos**

1. Concurrencia y paralelismo: Utilizar hilos de Java para modelar:
  - Operadores que reciben llamadas

- Despachadores que asignan recursos
  - Ambulancias y equipos médicos
  - Sistema de monitoreo en tiempo real
2. Estructuras de datos concurrentes: Implementar estructuras de datos thread-safe para manejar colas de espera, estado de recursos, y asignaciones.
  3. Sincronización: Aplicar mecanismos de sincronización para prevenir race conditions y deadlocks cuando:
    - Se asignen o liberen recursos
    - Se actualice el estado de las emergencias
    - Se modifique la prioridad de los casos
  4. Comunicación entre hilos: Implementar mecanismos para que los diferentes componentes del sistema se comuniquen entre sí.
  5. Visualización: Crear una interfaz (consola o gráfica) que muestre en tiempo real:
    - Emergencias pendientes y su prioridad
    - Estado de los recursos (disponibles, en camino, ocupados)
    - Tiempos de respuesta y estadísticas

### **Puntos para evaluar**

1. Diseño de la solución: Arquitectura del sistema y patrones de diseño aplicados
2. Implementación de la concurrencia: Uso correcto de hilos, sincronización y estructuras concurrentes
3. Manejo de escenarios críticos: Cómo se resuelven los conflictos y se previenen problemas como deadlocks
4. Eficiencia: Tiempo de respuesta y utilización óptima de recursos
5. Código limpio: Organización, legibilidad y buenas prácticas
6. Creatividad: Soluciones innovadoras a los desafíos planteados

### **Entrega**

- Repositorio Git: Código fuente completo y documentado
- Video explicativo: Demostración del sistema en funcionamiento y explicación técnica de la implementación
- Documentación: Manual técnico que detalle la arquitectura, patrones de concurrencia utilizados y decisiones de diseño
- Trabajo en equipo: Contribuciones y roles de cada miembro del equipo