Lectores y escritores
Práctica 3.2 - PC

Miembros del grupo:
Carlos Giraldo
Ricardo Pragnell

Al estudiar el código del MonitorPrioridadEscritores de la última práctica podemos hacernos las siguientes preguntas:

- **¿Qué pasaría si un escritor que está dentro (es decir, que ha invocado entrarEscribir pero todavía no ha invocado salirEscribir) intenta entrar otra vez (es decir, invoca a entrarEscribir)?**

  Al entrar de nuevo a escribir se quedaría esperando al ticket. Al no liberar la base de datos, hay un bloqueo de todos los procesos.

- **¿Qué pasaría si, cuando un lector está dentro (es decir, ha invocado entrarLeer pero todavía no ha invocado salirLeer), un escritor intenta entrar (es decir, invoca entrarEscribir) y luego el lector en cuestión intenta entrar otra vez (es decir, invoca a entrarLeer)?**

  Ocurriría como en el caso anterior, todos los hilos se quedarían esperando al ticket para entrar a la base de datos.

- **¿Por qué es interesante dejar que un hilo pueda entrar varias veces?**

  Para garantizar la estabilidad del sistema ante situaciones imprevistas, comportamientos erráticos y evitar posibles vulnerabilidades. Si estuviésemos simulando una red de ordenadores donde cada lector y escritor es un usuario, hay que tener todas las posibilidades cubiertas garantizando un servicio “estable”.

Ejercicio 1: Reentrante en lectura

Para contabilizar las veces que entra un hilo a leer empleamos un **HashMap**. Esto nos sirve para comprobar que un hilo tiene permiso de lectura. Cuando un lector sale se disminuye el número de veces que ha entrado en la tabla.

Ejercicio 2: Reentrante en escritura

Empleamos una variable entera para contabilizar el número de veces que entra un proceso escritor al sistema, y una variable **Thread** para la comprobación que permite a un escritor volver a entrar.

Ejercicio 3: Reentrante de lectura a escritura

En este apartado modificamos el código para permitir escribir a un lector si es el único lector con acceso. El problema de esta condición es que produce interbloqueos, ya que los lectores pueden quedar en espera del resto de lectores para obtener acceso de escritura. Por tanto hay que restringir la condición. como indica la pista del enunciado, a sólo permitir a un lector escribir si es el resto de lectores con “ganas” de escribir están en espera.

Para ello hacemos uso de un segundo HashMap, el primero (el mismo de la parte 1 y 2) se ocupa de almacenar las veces que entran a leer cada hilo. El segundo HashMap obtiene del primero los threads que después de leer intentan escribir y quedan (o no) en espera de escritura.

Ejercicio 4: Reentrante de escritura a lectura

Modificamos la condición de entrada a leer para adaptarse a la nueva situación. Ahora un hilo que quiera entrar a leer desde escritura debe de estar ya manipulando el recurso con anterioridad o esperar a que este se libere.