Sistemas Operativos II

Práctica 3 – Programación Multihilo

Buscador SSOOIIGLE (PARTE II)

Javier Alonso Albusac Jiménez
Escuela Superior de Informática :: Universidad de Castilla-La Mancha

Esta práctica pretende ser una continuación/ampliación de la práctica anterior. Si en la práctica dos se realizaba una única búsqueda mediante el uso de diferentes hilos, en esta se pretende simular la búsqueda simultánea de varios usuarios en un sistema operativo multiusuario.

En esta nueva versión del buscador SSOOIIGLE será necesario modelar usuarios (clientes) como entidades activas. Al no existir personas físicas que interactúen con el sistema, será necesario emularlos mediante la creación de hilos o procesos. En otras palabras, cada usuario será un hilo o proceso (decisión que deberá tomar el estudiante) que solicite al buscador la realización de una búsqueda.

El sistema deberá disponer de un diccionario de palabras[1]. En el momento de la creación de un usuario, se obtendrá al azar una de las palabras de este diccionario, y será la empleada para la búsqueda. Además de la palabra a buscar, el usuario debe tener asociado un identificador, que lo diferencia del resto, y una categoría o perfil:

- Usuario con cuenta gratuita.
- Usuario premium con límite de saldo. En este caso, el usuario debe tener asociada una cantidad de saldo, medida en créditos.
- Usuario premium con saldo ilimitado.

El perfil de un usuario determinará sus privilegios. Con la creación de los primeros usuarios deben comenzar las primeras búsquedas y, al mismo tiempo que éstas se están realizando, la creación de nuevos usuarios debe continuar, hasta un número determinado. Por ejemplo, podría probarse el sistema con la creación de 50 usuarios, aunque este valor debe ser configurable. Se recomienza dejar un tiempo prudencial en la creación entre clientes para conseguir la existencia simultánea de varios de ellos y proporcionar un nivel de concurrencia adecuado para evaluar la práctica.

El servidor de búsquedas debe dar preferencia a los clientes premium en la siguiente relación: 80% usuarios premium – 20 % usuarios con cuenta gratuita. Esto significa que, de 10 búsquedas realizadas, 2 deben ser para clientes con cuenta gratuita. No se debe producir inanición en clientes con cuenta gratuita. Es decir, se debe atender a este tipo de clientes (en menor medida) a pesar de que existan clientes premium.

El servicio de búsqueda debe ser concurrente, pero debe está limitado a N replicas. Esto quiere decir que si N = 4, tan solo se podrán atender a cuatro clientes de forma simultánea. El resto deberá esperar a que llegue su turno. Si en la práctica anterior se realizaba búsquedas sobre una única fuente o libro (archivo .txt), en esta ocasión cada búsqueda se debe realizar sobre el conjunto de libros que estén

[1] El diccionario podría ser un array de *strings* con algunas palabras definidas de forma manual por el programador. Otra alternativa es procesar uno de los libros (archivos.txt) para extraer palabras con más de dos caracteres e incluirlas en un array, todo ello de forma automática.

incluidos como fuentes. Además, en la práctica anterior se utilizaban varios hilos para un libro. Para la práctica 3 se pide que la búsqueda se haga de forma simultánea en las diversas fuentes, lo que implica que al menos exista un hilo para cada libro. Es opcional y mejora la calificación que el estudiante consiga emplear varios hilos para la búsqueda en cada fuente.

El tipo de cliente también condiciona la búsqueda de la siguiente manera:

- **Cliente con cuenta gratuita**: la búsqueda está limitada a un número fijo de palabras. Si la búsqueda ha superado ese valor límite, se debe dar por finalizada la búsqueda haciendo llegar al cliente los resultados obtenidos hasta ese momento. Se debe tener especial cuidado en el control de palabras, en el caso de emplear varios hilos para la búsqueda.
- **Cliente con cuenta premium con límite de saldo**: cada palabra encontrada supone el coste de 1 crédito. Cuando el saldo del cliente llega a cero, la búsqueda debe detenerse momentáneamente para solicitar al cliente un nuevo pago y recarga de saldo. Una vez hecho esto, la búsqueda debe continuar. Se debe tener especial cuidado en el control de saldo, en el caso de que se empleen varios hilos para la búsqueda. Más adelante se hablará con más detalle del sistema de pago.
- **Cliente con cuenta premium ilimitada**: no tiene restricciones en la búsqueda.

El sistema de pago es un servicio que debe ser modelado mediante un hilo o proceso. Este servicio es único, lo que quiere decir que no puede ser empleado al mismo tiempo por más de un cliente. El hilo/proceso debe permanecer a la espera/bloqueado, hasta que un cliente solicite sus servicios, que será cuando se agote su saldo. Este servicio no debe ser anónimo, el sistema de pago debe conocer la identidad del cliente, y el servicio debe notificar al cliente que su saldo ha sido modificado cuando se haya efectuado el pago. Si una búsqueda se ha detenido por falta de saldo, se deberá reanudar una vez restaurado.

Cuando finaliza una búsqueda se deben hacer llegar los resultados al cliente que hizo la solicitud (y que permanece a la espera), para que éste sea quien muestre por pantalla los resultados o bien los registre en un archivo. El modo en el que deben mostrarse los resultados es exactamente el mismo al de la segunda práctica, pero, además, se deberá medir el tiempo total de búsqueda dedicado al cliente e incluirlo en los resultados.

Por último, se debe mostrar por pantalla la finalización de cada uno de los clientes y se debe comprobar que sean tantos como se crearon inicialmente.

ATENCIÓN: en la siguiente página se incluye la autoevaluación que deberá ser completada por los estudiantes. Se recomienda su lectura antes de comenzar a realizar la práctica.

Autoevaluación Práctica 3

Sistemas Operativos II

Escuela Superior de Informática
Universidad de Castilla-La Mancha

Nombre y Apellidos del estudiante:

Buenos hábitos de programación	
Ítem.	**¿Se cumple?**
El código de la práctica está dividido en varios archivos y directorios	☑
Los nombres de variables y funciones son representativos	☑
Los nombres de constantes se han representado en mayúsculas	☑
El código está correctamente tabulado y alineado	☑
Se inicializan las variables creadas, preferiblemente en el momento de creación	☑
El código está escrito en su totalidad en inglés	☐
Se han utilizado variables globales	Sí | No
En el caso de haber utilizado variables globales, se utiliza en el nombre un prefijo distintivo que permite tomar conciencia rápidamente de que se trata de una variable global.	☑
En el caso de haber utilizado punteros, también se emplea una convención para los nombres.	☐
No se incluyen valores numéricos directamente en estructuras condicionales y de control (bucles). En su lugar, se emplean variables.	☑
Las funciones tienen un propósito muy concreto	☑
Se ha evitado excesivo anidamiento de estructuras condicionales dentro de una misma función.	☑
No se repiten fragmentos de código dentro del proyecto	☑
El código, dentro de un mismo archivo, está correctamente estructurado.	☑
Utilizo comentarios en mi código	☑
Se han incluido comentarios de cabecera en cada archivo con información útil sobre el contenido.	☑
Se han eliminado el mayor número de dependencias posibles en la interacción con el usuario	☑
He utilizado patrones de diseño	☐
He utilizado un repositorio con control de versiones	✓
He automatizado la compilación del proyecto con herramientas como make	☑
Considero que el código de mi proyecto está bien testeado	☑
He utilizado un depurador durante el desarrollo de la práctica para detectar los errores	☑

El código es robusto y responde bien ante posibles excepciones	☐
En el código hago un control de posibles errores que se puedan producir	☑
Se hace uso de control de excepciones en C++ *try* y *catch*	☐
Los mensajes son mostrados a través de la salida estándar y los errores son redireccionados a la salida estándar de errores.	☑
Cumplimiento de objetivos	
Ítem.	**¿Se cumple?**
Creación de clientes correcta	☑
Se continúa creando clientes mientras se realizan búsquedas	☑
Existe concurrencia entre clientes, buscador y sistema de pago	☐
Clientes creados que no pueden ser atendidos permanecen a la espera	☑
Existe un diccionario de palabras para elegir la búsqueda	☑
El diccionario se genera de forma automática	☑
Existe una clase cliente para encapsular toda la información relativa a éste.	☑
Se han creado estructuras de datos para encapsular los resultados de búsqueda para cada cliente	☑
La atención entre clientes premium y clientes con cuenta gratuita sigue la relación 80-20.	☐
El servicio de búsqueda está limitado a N réplicas	☑
Control del número total de hilos a través de *hardware_concurrency()*	☐
Para cada cliente se hace la búsqueda en todas las fuentes/libros de forma simultánea	☑
Se utiliza al menos un hilo para la búsqueda en cada fuente y cada cliente.	☑
[opcional] Se utilizan varios hilos por fuente para cada cliente.	☐
Se mide el tiempo total de búsqueda para cada cliente	☐
Para los clientes con cuenta gratuita se controla el limite de palabras y se detiene la búsqueda en caso de llegar a ese límite	☑
Control adecuado del saldo en clientes premium con saldo limitado	☑
Se ha incluido un sistema de pago de uso exclusivo	☑
El sistema de pago no es anónimo	☐
Las búsquedas detenidas por falta de saldo se reanudan correctamente tras la realización del pago	☑
Se hace llegar al cliente el resultado de la búsqueda	☑
El cliente muestra por pantalla o guarda en un archivo los resultados de la búsqueda	☑
Se muestra por pantalla la finalización de cada cliente.	☑
Se han utilizado patrones de sincronización entre procesos/hilos	☑
Se ha estudiado la eficiencia del sistema y ha influido en el diseño de estructura de datos y la estrategia de creación de hilos/procesos	☐
¿Con que nota entre 0 y 10 calificarías tu práctica? (obligatorio responder)	Entre 7 y 7,5