Práctica final: Tsunami Democrático Cultural Leonés

Miguel Ángel Conde González
Antonio Gómez García
Mario Enrique Casado García

4 de diciembre de 2019

Definición del problema

El objetivo de la práctica es realizar un programa que simule una aplicación social para la convocatoria de reuniones democrático-culturales pacíficas en León. A dicha aplicación se accedera por invitación o lectua de un QR y tiene que ser aprobado por alguno de los miembros seniors de la organización (threads). La práctica tiene una parte básica donde se define el funcionamiento básico del sistema y que es obligatorio implementar (supondrá como mucho el 80 % de la nota) y, además, se proponen una serie de mejoras opcionales para aumentar la nota (como mucho supondrá el 20 % de la nota).

Para aprobar la práctica es necesario que la parte básica funcione correctamente. La nota se asignará en base a la calidad del código entregado, valorándose:

- Política de nombres (coherencia en el nombrado de variables y funciones).
- Sangrado (indentación).
- Comentarios (cantidad, calidad y presentación).
- Legibilidad: Nombre de variables, funciones, etc.
- Reusabilidad y mantenibilidad: Uso de parámetros, etc.

La práctica es en grupo y se evaluará según la metodología CTMTC explicada en clase.

Parte Básica (80 % de la nota)

La aplicación funciona de la siguiente manera:

- Los usuarios para poder emplear la aplicación necesitan que se acepte su solicitud por personal senior de la misma, esto supone entrar en una cola de espera para que la solicitud sea aceptada y se pueda comenzar a usar.

- La cola de solicitudes tiene un máximo de 15 usuarios, si se supera esa cifra la solicitud será rechazada. Para atender estas solicitudes se cuenta con 3 usuarios uno para las solicitudes por invitación, otro para las solicitudes por QR y un tercero que puede procesar ambos tipos de solicitudes (atendedor PRO) y que se dedica a unas u otras indistintamente. Esto implica que 3 solicitudes pueden ser atendidas mientras las otras 12 esperan.
- Un 30 % de las solicitudes por QR se descartan por no considerarse muy fiables y un 10 % de las solicitudes por invitación lo hacen por cansarse de esperar. Además La aplicación no está muy bien hecha y descarta solicitudes de forma aleatoria y sin comunicarlo, así que del porcentaje restante de solicitudes encoladas se descarta un 15 %.
- A cada solicitud se le asignará un identificador único y secuencial (solicitud_1, solicitud_2,... solicitud_N) a medida que vayan comenzando el proceso.
- Una vez una solicitud haya sido aceptada esta puede vincularse a una inciativa cultural. Para que una iniciativa siga adelante es necesario tener a 4 usuarios. Las iniciativas las gestiona un coordinador cultural. Que una vez estén los 4 solicitantes listos comienza la actividad que dura 3 segundos.
- Si ya se tienen 4 solicitudes asociadas a una actividad cualquier nueva solicitud que quiera vincularse a la misma requiere que finalicen todas las otras y por tanto dejen hueco. Hasta que no finalicen las 4 solicitudes no podrá accederse a la espera por una nueva actividad y se seguirá ocupando espacio en la cola de solicitudes.

Aspectos a tener en cuenta:

- El personal de atención a solicitudes tiene un identificador único (atendedor_1, atendedor_2, atendedor_3).
- Solo cuando una solicitud haya sido atendida podra pasar a asociarse a una actividad cultural.
- Las solicitudes comprueban cada 3 segundos si han sido descartadas según alguna de las posibles casúisticas y en caso de ser así abandonan la cola.
- Cada vez que se atienden 5 solicitudes, el atendedor descansa 10 segundos, con lo que nadie puede acceder al puesto hasta que vuelve de su descanso.
- Las solicitudes de cada tipo deben atenderse por orden de llegada.
- Si el atendedor de invitaciones o el de qrs no tiene solicitudes que atender puede tratar una del otro tipo, pero una vez termine de hacerlo deberá comprobar de nuevo sus invitaciones y ver si hay alguna de las que le corresponde atender.
- El atendedor pro no dicriminaría si hay más de uno u otro y siempre usa el criterio de antigüedad.

- De las solicitudes a atender, el 70 % tiene todo en regla, el 20 % tiene errores en los datos personales y el 10 % tiene vinculada una ficha con antecedentes policiales. Esto tiene una implicación en los tiempos de atención:
  - 70 % atención correcta – En estos casos, el tiempo de espera está entre 1 y 4 segundos y después se puede solicitar o no participar en una actividad cultural.
  - 20 % errores en datos personales – En estos casos, el tiempo de espera está entre 2 y 6 segundos y después se puede solicitar o no participar en una actividad cultural.
  - 10 % antecedentes – En estos casos, el tiempo de espera está entre 6 y 10 segundos y no puede unirse a una actividad cultural, luego después del tiempo de espera debe dejar su sitio en la cola.
- Cuando la atención ha sido correcta o el error era solo en los datos personales, la solicitud podrá asociarse o no a una actividad cultural (50 % de probabilidad). Si no se asocia a la actividad la solicitud deja el sistema, en caso de querer asociarse se libera el hueco en la cola de atención y se ocupa uno en la cola de la actividad cultural (si esto es posible).

Toda la actividad quedará registrada en un fichero plano de texto llamado `registroTiempos.log`. En concreto, es necesario registrar al menos:

- Cada vez que una solicitud accede al sistema
- Cada vez que una solicitud deja el sistema por no ser fiable o por cansarse de esperar a ser atendida.
- Cada vez que una solicitud termina correctamente cuánto ha tardado.
- Cada vez que es atendida una solicitud con los datos personales incorrectos y cuanto ha tardado.
- Cada vez que una solicitud ha abandonado el sistema por antecedentes.
- Cuando una solicitud solicita la vinculación a una actividad cultural.
- Cuando una solicitud se vincula a la actividad cultural.
- Cuando una solicitud finaliza una actividad cultural.
- Se registra el inicio y final del descanso de cada atendedor.
- Al finalizar el programa se debe terminar de atender todas las solicitudes pendientes pero ya no podrán vincularse a una actividad cultural.

Consideraciones prácticas:

- Simularemos el inicio de funcionamiento del sistema mediante señales. En caso de que una solicitud por invitación quiera acceder al sistema se usará la señal SIGUSR1 y en el caso de ser por QR se enviará la señal SIGUSR2. Cada vez que se le envíe la señal, supone que ha accedido una solicitud nueva al sistema.

- Es obligatorio el uso de mensajes que se escribirán en un log y se mostrarán por pantalla. El formato de tales mensajes será:

  `[YYYY-MM-DD HH:MI:SS] identificador: mensaje`

  Donde `identificador` puede ser el identificador del usuario, el identificador del facturador y `mensaje` es una breve descripción del evento ocurrido.

- Las entradas del log deben quedar escritas en orden cronológico.

- El programa finaliza cuando recibe la señal SIGINT y deberá hacerlo correctamente.

Partes opcionales (20 % de la nota)

- Asignación estática de recursos (10 %):
  - Modifica el programa para que el número de solicitudes que pueden tratarse en el sistema sea un parámetro que reciba el programa al ser ejecutado desde la línea de comandos.
  - Modifica el programa para que el número de atendedores que atienden cualquier tipo de solicitud sea un parámetro que reciba el programa al ser ejecutado desde la línea de comandos.

- Asignación dinámica de recursos I (5 %):
  - Modifica el programa para que el número solicitudes que pueden acceder al sistema pueda modificarse en tiempo de ejecución.
  - Solamente es necesario contemplar un incremento en solicitudes. No es necesario contemplar la reducción.
  - Cada vez que se cambie el número de solicitudes tiene que reflejarse en el log.

- Asignación dinámica de recursos II (5 %):
  - Modifica el programa para que el número de atendedores se pueda modificar en tiempo de ejecución.
  - Solamente es necesario contemplar un incremento en el número de atendedores. No es necesario contemplar la reducción.
  - Cada vez que se produce un cambio en este sentido debe quedar reflejado en el log.

Escritura de mensajes en log

Es recomendable utilizar una función parecida a esta para evitar repetir líneas de código. Recibe como parámetros dos cadenas de caracteres, una para el identificador y otra para el mensaje (la fecha la calcula la propia función):

```
1  void writeLogMessage(char *id, char *msg) {
2    // Calculamos la hora actual
3    time_t now = time(0);
4    struct tm *tlocal = localtime(&now);
5    char stnow[19];
6    strftime(stnow, 19, "%d/%m/%y %H:%M:%S", tlocal);
7
```

```
8   // Escribimos en el log
9   logFile = fopen(logFileName, "a");
10  fprintf(logFile, "[%s] %s: %s\n", stnow, id, msg);
11  fclose(logFile);
12 }
```

Ejemplo 1: Diseño de la parte básica