DSD P2

PABLO BLANCO LOPEZ

Marzo 2024

1 Introducción

La práctica 2 de Desarrollo de Sistemas Distribuidos consta de hacer un sistema cliente-servidor de una calculadora con llamadas a procedimiento remoto (RPC). Se pide hacer que la calculadora haga operaciones básicas, y como puntos extra que haga operaciones con cierta complejidad. He elegido las operaciones básicas, entre matrices y entre vectores.

2 Funcionamiento

Una vez lanzado el servidor, si se ejecuta el cliente salta un menú donde pide introducir por consola un número del 1 al 4, siendo el siguiente menú:

1. Operaciones de flotantes
2. Operaciones de matrices
3. Operaciones de vectores
4. Salir

Dentro del menú 1. Operaciones de flotantes, pide introducir el primer valor numérico de la operación, después el operador, siendo estos suma, resta, multiplicación y división, + - * /, respectivamente, procediendo del segundo valor numérico. Una vez realizada la operación, preguntará si quiere realizar otra operación, para responder si escriba 'y', para responder no escriba 'n'. En el caso de querer otra operación, procedería igual que anteriormente se ha descrito, pero preguntará antes de escribir los valores numéricos si quiere que ese valor sea el resultado de la operación anterior, en caso de que no, podrá introducir el valor por terminal.
Dentro del menú 2. Operaciones de matrices, pedirá primero ingresar el tamaño de las matrices, siendo este tamaño el de las dos matrices, acto seguido pide introducir los valores de la primera matriz, el operador, siendo estos suma, resta, multiplicación entre matrices, y multiplicación esclar, + - * x, respectivamente. En el caso de seleccionar uno de los tres primeros, pide introducir la segunda matriz, pero al seleccionar el producto escalar, pide introducir un escalar para

que multiplique la matriz.
Por último, en el menú 3. Operaciones de vectores, es parecido al menú anterior en cuanto al funcionamiento, pide introducir el tamaño de los vectores, seguido del primer vector, la operación, siendo estas sumar, restar, producto escalar y producto vectorial, + - * x, en el caso del último solo se puede con vectores de tamaño tres. Después pide introducir el segundo vector de la operación

3 Explicación

Para realizar la práctica he empezado con definir mi archivo cal.x, añadiendo aquí el programa principal con sus diferentes versiones, la pimera versión implementada es la que hace operaciones de flotantes, la segunda, que hace operaciones de matrices y por último la tercera, que hace operaciones de vectores, cada una con un struct pasado como parámetro definido con los diferentes valores que hacen falta para la operación, así como el primer valor de la operación, ya sea un flotante, una matriz o un vector, el operador, el segundo valor de la operación, o el tamaño de matrices o vectores.
Para continuar hay que escribir por consola el siguiente comando para que RPC genere los diferentes archivos donde se trabaja.

```
$ rpcgen -NCa cal.x
```

Esto genera los archivos *cal_client.c*, *cal_server.c*, *cal_clnt.c*, *cal_svc.c*, *cal_xdr.c*, *cal.h* y *Makefile.cal*. Se trabajará únicamente con los archivos *cal_client.c* y *cal_server.c*, que son los archivos del cliente y del servidor respectivamente. En *cal_client.c* se ubica el código de los menús y de los inputs de los diversos valores numéricos de las operaciones, flotantes, matrices y vectores, y es realmente en el servidor, en el archivo *cal_server.c* dónde se calcula el resultado de cada operación, devolviendo un struct que tiene un número de error y una unión de resultados, que dependiendo de cada llamada a cada servidor nos devuelve un flotante, una matriz o un vector.
Durante el desarrollo de la práctica, tuve ciertos problemas sobre todo a la hora de declarar las matrices en *cal.x*, ya que hay que hacer un vector de vectores y no recordaba bien como hacerlo, además cuando rpcgen te genera el resto de archivos, lo traduce a C y para un vector crea un struct con el puntero al tipo de dato del vector y el tamaño del vector, una vez aclarado esto, mi problema fue que para pasarle al servidor un argumento del tipo de dato de un vector hay que rellenar también el tamaño del vector, ya que si no lo haces te salta el error 'can´t encode arguments', pero una vez que me di cuenta lo solucioné relativamente fácil. Por su puesto hay que alojar memoria a las matrices y a los vectores, y esto también me dió ciertos problemas, al principio empecé con un 'malloc' y el tamaño del vector (también aplicable a las matrices), pero lo que hace 'malloc' es alojar una cierta cantidad de bytes al puntero que corresponde, por lo que lo que tuve que hacer es multiplicar el tamaño del vector por la cantidad del tipo de dato del vector, obteniendo esa cantidad de bytes con la

función 'sizeof()'. Una vez realizada la asignación de memoria ya se puede pedir al cliente que introduzca los diferentes valores que quiere que tenga el vector.

4 Entrega

Para la entrega he comprimido mi carpeta de trabajo, que contiene principalmente los tres archivos con los que he trabajado, *cal.x*, *cal_client.c* y *cal_server.c*, y después una carpeta llamada 'test' que es donde realmente compilo y ejecuto todo.