

RPC: Operaciones básicas con vectores dinámicos

Alumno: Raúl Rodríguez Pérez

Grupo: DSD2

1. Introducción y explicación del .x

Para la realización de la práctica 2 de la asignatura de Desarrollo de sistemas distribuidos he desarrollado, a parte de la calculadora básica, las operaciones básicas (+,-,/,*) con el uso de vectores dinámicos. Todo ello ha sido realizado bajo la implementación de programas distribuidos con llamadas a procedimientos remotos RPC.

Para la ejecución de este ejercicio lo primero que he realizado ha sido el .x, en él describo la estructura que voy a utilizar como vector, y las funciones o métodos que va a desarrollar el servidor:

```
rpc_vectores > ≡ vectores.x
1  struct miVector{
2  |    float vector<>;
3  };
4
5  program VECTORES_PROG{
6  |    version VECTORES_VER{
7  |        miVector SUMAR(miVector,miVector)=1;
8  |        miVector RESTAR(miVector,miVector)=2;
9  |        miVector DIVIDIR(miVector,miVector)=3;
10 |        miVector MULTIPLICAR(miVector,miVector)=4;
11 |    }=1;
12 }=0x200000001;
13 |
```

En primer lugar, tuve que leer varios repositorios sobre RPC, y sobre los tipos de datos de XDR y sus equivalencias en C. Finalmente encontré este documento:

- http://ocw.uc3m.es/ingenieria-informatica/sistemas-distribuidos-2013/Tema6_II/amadasaprocedimientosremotos.pdf

En él se profundiza en gran medida sobre RPC, y justo en la pág 94 encontré como crear la estructura de mi vector:

Principales tipos de datos en XDR

Vectores de tamaño variable:

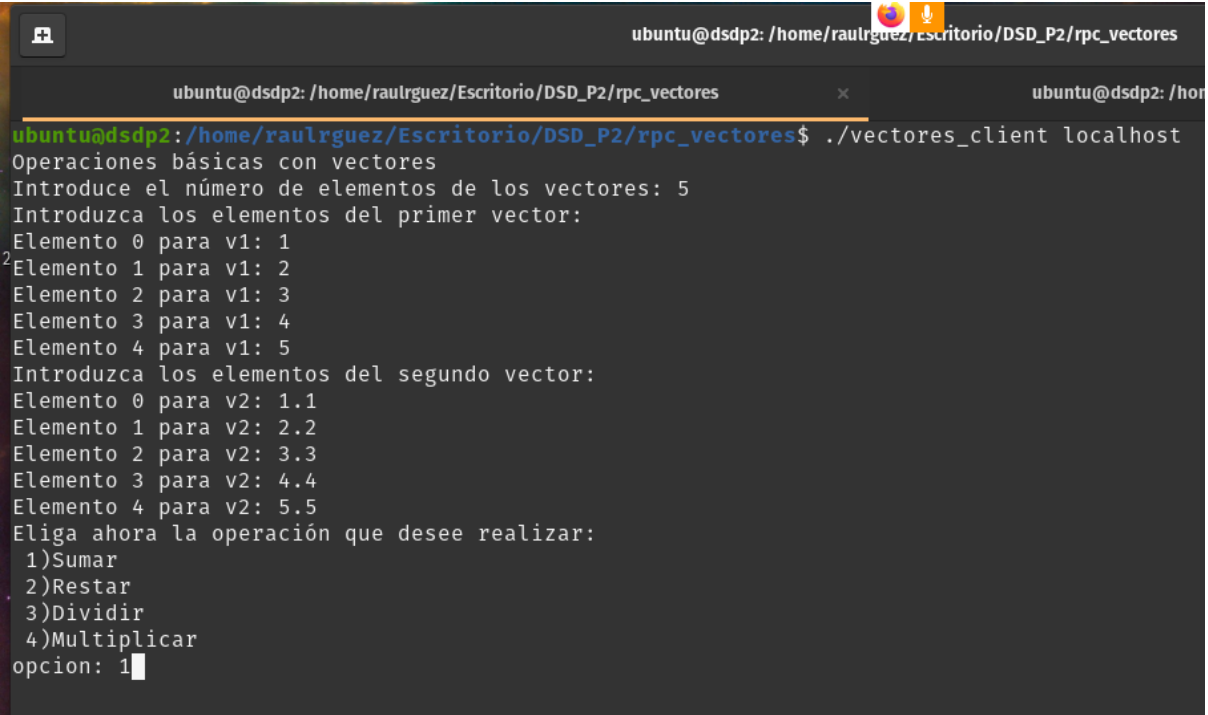
Declaración: int a<12>;
 float b<>;

Equivalente en C: struct {
 int a_len;
 int *a_val;
 } a;
 struct {
 int b_len;
 float *b_val;
 } b;

Luego simplemente, cree las funciones del servidor, en las cuales le paso 2 estructuras de “miVector”, lo que sería equivalente a pasarle los 2 vectores por parámetros.

2. Caso de ejecución y detalles del código

Al ejecutar el código, lo primero que debemos de hacer es introducir por teclado el número de elementos que queremos que tengan los dos vectores. Tras esto, se procede a rellenar dichos vectores (los elementos son “float” por lo que podemos poner números decimales). Por último nos pide elegir cuál es la operación que queremos realizar con dichos vectores.



```
ubuntu@dspd2: /home/raulrguez/Escritorio/DSD_P2/rpc_vectores
ubuntu@dspd2: /home/raulrguez/Escritorio/DSD_P2/rpc_vectores
ubuntu@dspd2:/home/raulrguez/Escritorio/DSD_P2/rpc_vectores$ ./vectores_client localhost
Operaciones básicas con vectores
Introduce el número de elementos de los vectores: 5
Introduzca los elementos del primer vector:
Elemento 0 para v1: 1
Elemento 1 para v1: 2
Elemento 2 para v1: 3
Elemento 3 para v1: 4
Elemento 4 para v1: 5
Introduzca los elementos del segundo vector:
Elemento 0 para v2: 1.1
Elemento 1 para v2: 2.2
Elemento 2 para v2: 3.3
Elemento 3 para v2: 4.4
Elemento 4 para v2: 5.5
Elija ahora la operación que desee realizar:
1)Sumar
2)Restar
3)Dividir
4)Multiplicar
opcion: 1
```

Una vez hemos realizado todos estos pasos, se hace la llamada al programa: “vectores_prog_1 (host, var1, oper, var2);”. Dicho programa tiene como parámetro 2

estructuras `miVector`, y la operación que hay que realizar. Gracias a este último parámetro, y con la ayuda de un `switch`, se procede a la llamada de una de las distintas funciones que posee el servidor para realizar las operaciones básicas.

```
vectoros_prog_1(char *host, miVector var1, char oper, miVector var2)
{
    CLIENT *clnt;
    miVector *result;
    char *operacion;

#ifdef DEBUG
    clnt = clnt_create (host, VECTORES_PROG, VECTORES_VER, "udp");
    if (clnt == NULL) {
        clnt_pcreateerror (host);
        exit (1);
    }
#endif /* DEBUG */

    switch (oper){
        case '+':
            result = sumar_1(var1, var2, clnt);
            operacion = "suma";
            break;
        case '-':
            result = restar_1(var1, var2, clnt);
            operacion = "resta";
            break;
        case '/':
            result = dividir_1(var1, var2, clnt);
            operacion = "divicion";
            break;
        case '*':
            result = multiplicar_1(var1, var2, clnt);
            operacion = "multiplicacion";
            break;
    }
}
```

En las funciones del servidor se encuentra el código para realizar las operaciones básicas, además del control de errores para los casos, como por ejemplo, de que surgiera la división por 0. Además cabe destacar que en dichas funciones también se realiza la liberación de memoria de los vectores, justo antes de nada, para liberar lo que se ha usado anteriormente. Una vez hecho el cálculo se envía la estructura con la solución y esta se imprime por pantalla.

```
opcion: 1
El resultado de la suma de vectores es:
(2.100000, 4.200000, 6.300000, 8.400000, 10.500000)
ubuntu@dspd2:/home/raulrguez/Escritorio/DSD_P2/rpc_vectores$
```