



UNIVERSIDAD
DE GRANADA

PRÁCTICA 2: APACHE THRIFT

**MEMORIA DE LA SOLUCIÓN DE LA PRÁCTICA 2-B DE DESARROLLO
DE SISTEMAS DISTRIBUIDOS**

*Implementación de una calculadora con
llamadas a procedimientos remotos.*

AUTOR: ÁLVARO LÓPEZ VERGARA

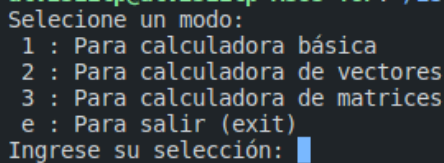
11 abril 2024

1. Introducción:

En esta memoria vamos a documentar la implementación de una calculadora con *apache thrift*, de nuevo con una estructura cliente-servidor, como en la parte uno de la práctica. Trataremos distintos puntos como: la implementación, las funcionalidades de la calculadora y cómo usarla; y mostraremos algunas pruebas con capturas de pantalla sobre la ejecución.

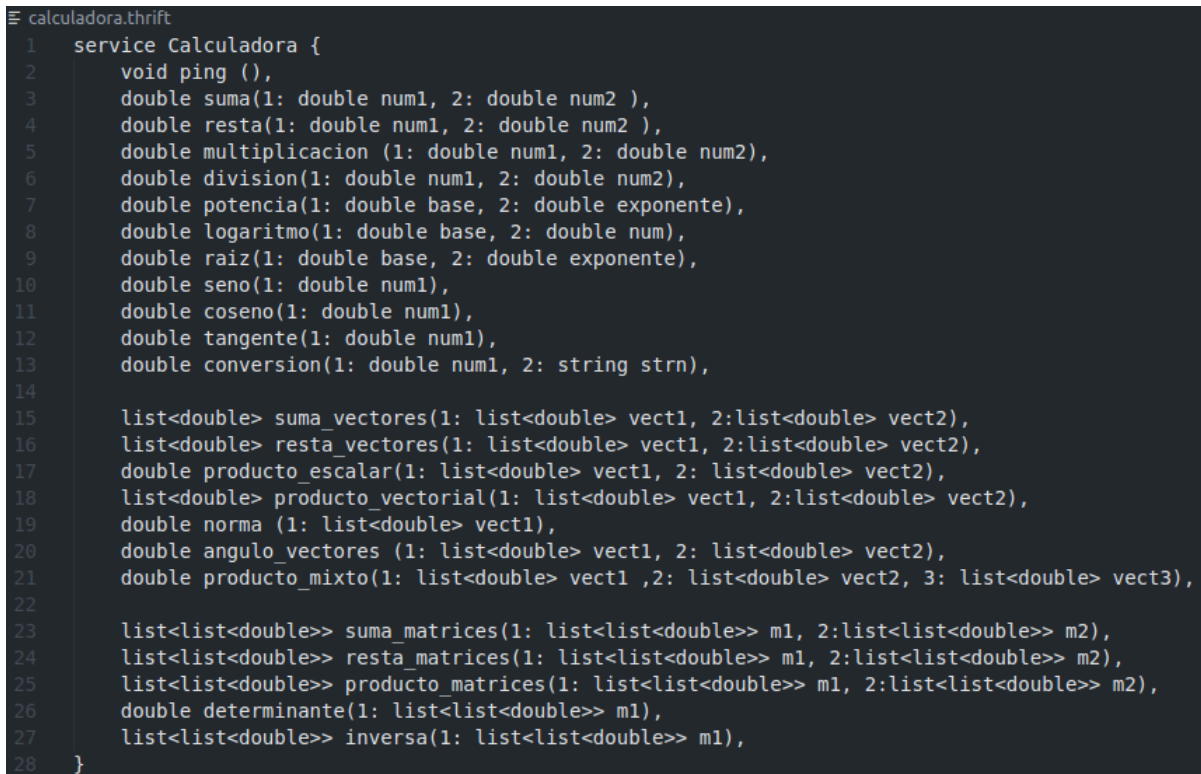
2. Implementación y desarrollo:

Se han implementado 3 calculadoras diferentes en esta ocasión, una calculadora básica que realiza operaciones matemáticas sencillas y trigonométricas. Otra de vectores, que realiza operaciones más complejas usando estos como argumentos. Y finalmente una calculadora de matrices para realizar distintas operaciones con ellas.



```
Seleccione un modo:
1 : Para calculadora básica
2 : Para calculadora de vectores
3 : Para calculadora de matrices
e : Para salir (exit)
Ingrese su selección: █
```

Inicialmente hemos declarado las operaciones en nuestro archivo *thrift*, tal como se muestra en la siguiente imagen:



```
calculadora.thrift
1 service Calculadora {
2     void ping (),
3     double suma(1: double num1, 2: double num2 ),
4     double resta(1: double num1, 2: double num2 ),
5     double multiplicacion (1: double num1, 2: double num2),
6     double division(1: double num1, 2: double num2),
7     double potencia(1: double base, 2: double exponente),
8     double logaritmo(1: double base, 2: double num),
9     double raiz(1: double base, 2: double exponente),
10    double seno(1: double num1),
11    double coseno(1: double num1),
12    double tangente(1: double num1),
13    double conversion(1: double num1, 2: string strn),
14
15    list<double> suma_vectores(1: list<double> vect1, 2: list<double> vect2),
16    list<double> resta_vectores(1: list<double> vect1, 2: list<double> vect2),
17    double producto_escalar(1: list<double> vect1, 2: list<double> vect2),
18    list<double> producto_vectorial(1: list<double> vect1, 2: list<double> vect2),
19    double norma (1: list<double> vect1),
20    double angulo_vectores (1: list<double> vect1, 2: list<double> vect2),
21    double producto_mixto(1: list<double> vect1 ,2: list<double> vect2, 3: list<double> vect3),
22
23    list<list<double>> suma_matrices(1: list<list<double>> m1, 2: list<list<double>> m2),
24    list<list<double>> resta_matrices(1: list<list<double>> m1, 2: list<list<double>> m2),
25    list<list<double>> producto_matrices(1: list<list<double>> m1, 2: list<list<double>> m2),
26    double determinante(1: list<list<double>> m1),
27    list<list<double>> inversa(1: list<list<double>> m1),
28 }
```

Aquí declaramos todas las operaciones que realiza nuestra calculadora; y con el compilador de *thrift* generamos todo código intermedio entre cliente y servidor, de forma que nos abstraemos completamente del paso de mensajes intermedio para nuestro programa distribuido.

Por tanto solo implementamos nuestro servidor, donde se realizaron los cálculos y operaciones; y nuestro cliente, donde se realizarán llamadas a procedimientos remotos dirigidas al servidor.

Además, en *cliente.py*, tenemos el *main* principal con un menú implementado para seleccionar las operaciones y argumentos de estas por consola.

3. Funcionalidades y manual de usuario:

Los comandos de ejecución son:

python3 cliente.py para el cliente,

python3 servidor.py para el servidor.

Al ejecutar nuestro programa, se nos presenta inicialmente, el menú mostrado anteriormente:

```
Seleccione un modo:
1 : Para calculadora básica
2 : Para calculadora de vectores
3 : Para calculadora de matrices
e : Para salir (exit)
Ingrese su selección: █
```

Tras elegir una de las opciones 1, 2 o 3, se nos muestra el menú de cada calculadora, por ejemplo si elegimos el 1:

```
Ingrese su selección: 1
*****
Seleccione una operación:
+ : Para una suma
- : Para una resta
* : Para una multiplicacion
/ : Para una division
^ : Para una potencia
l : Para un logaritmo
r : Para una raiz de exponente x
s : Para seno (grados)
c : Para coseno (grados)
t : Para tangente (grados)
v : Para una conversion (grados-radianes)
o : Para volver
Ingrese su selección: █
```

Al seleccionar una operación nos pedirá los argumentos necesarios para cada operación, para posteriormente mostrarnos el resultado; y de nuevo pedirnos otra nueva operación a realizar.

```
o : Para volver
Ingrese su selección: /
Introduzca el primer número: 1864
Introduzca el segundo número: 4
~~ El resultado de la operación / es: 466.0 ~~
*****
Seleccione una operación:
+ : Para una suma
- : Para una resta
```

La implementación tiene cierta seguridad, por ejemplo si no introducimos alguna de las operaciones válidas, o si intentamos dividir por 0:

```
o : Para volver
Ingrese su selección: /
Introduzca el primer número: 5
Introduzca el segundo número: 0
Error. Introduzca un divisor distinto de cero.
Introduzca el segundo número: 3
~~ El resultado de la operación / es: 1.6666666666666667 ~~
*****
Seleccione una operación:
+ : Para una suma
```

```
e : Para coseno (grados)
t : Para tangente (grados)
v : Para una conversion (grados-radianes)
o : Para volver
Ingrese su selección: 8
Operación no válida.
Seleccione una operación:
+ : Para una suma
```

Por otra parte la calculadora de vectores tiene las siguiente operaciones:

```
0 : Para salir (exit)
Ingrese su selección: 2
*****
Vector 1: []
Vector 2: []
Vector 3: []
Seleccione una operación:
q : Para modificar algún vector
+ : Para una suma de vectores
- : Para una resta de vectores
. : Para producto escalar
x : Para producto vectorial
n : Para norma
a : Para angulo entre vectores
m : Para producto mixto
o : Para volver al modo anterior
Ingrese su selección: █
```

Al igual que en la parte 1 de la práctica, además de mostrarnos las operaciones, nos mostrará los 3 vectores que tenemos almacenados. Estos son los vectores que usamos como argumentos para las operaciones, para poder modificarlos dinámicamente usamos la letra q, entonces nos pedirá el vector a sobrescribir, y luego sus valores. Gracias a esto, podemos hacer cualquier combinación en las operaciones, por ejemplo, $v_3 - v_1$ o $v_2 \times v_3$.

Concretar, que hay ciertas operaciones que sólo pueden realizarse sobre vectores de dimensión 3, pues no tienen sentido matemático con dimensión superior, y que para una sola operación estos deben tener misma dimensión, por la misma razón. Estas operaciones son: *producto_vectorial* y *producto_mixto*.

Ejemplo de introducción de los vectores:

```
0 : Para volver al modo anterior
Ingrese su selección: q
Selecciona un vector a modificar (1, 2 o 3): 1
Introduce los elementos del vector, separados por espacios:
2 5 6
*****
Vector 1: [2.0, 5.0, 6.0]
Vector 2: []
Vector 3: []
```

Destacar que puede introducirse cualquier longitud de vector, y que para introducir los elementos tiene que hacerse separados por espacios, y sin espacio al final. (2' '5' '6)

Para seleccionarlos en las operaciones es de la misma manera:

```
0 : Para volver al modo anterior
Ingrese su selección: +
Selecciona dos vectores (1, 2 o 3) separados por espacios: 1 1
~~ El resultado de la operación + es: [4.0, 10.0, 12.0] ~~
*****
Vector 1: [2.0, 5.0, 6.0]
Vector 2: []
```

Finalmente la calculadora de matrices, que nos presenta las siguientes operaciones implementadas:

```
Ingrese su selección: 3
*****
Matriz: [[], [], []]
Matriz: [[], [], []]
Seleccione una operación:
q : Para modificar alguna matriz
+ : Para suma de matrices (m1 + m2)
- : Para resta de matrices (m1 - m2)
x : Para producto de matrices (m1 x m2)
d : Para determinante (m1)
i : Para inversa (m1)
o : Para volver al modo anterior
Ingrese su selección: █
```

En este caso, el orden en el que se aplican las operaciones a las matrices está predefinido, es decir que para realizar $m_2 \times m_1$, tendremos que cambiar los valores de m_1 por los de m_2 y viceversa.

Tenemos en este caso 2 matrices a modificar, con las que realizamos las operaciones. Para introducir los valores de las matrices se hace de forma similar a los vectores:

```
o : Para volver al modo anterior
Ingrese su selección: q
Seleccione una matriz a modificar (1 o 2): 1
Introduce los elementos de la matriz, separados por espacios. Presiona Enter al finalizar.
Introduce una fila (deja vacío para terminar): 5 6 4
Introduce una fila (deja vacío para terminar): 1 2 8
Introduce una fila (deja vacío para terminar): 0 7 5
Introduce una fila (deja vacío para terminar):
*****
Matriz: [[5, 6, 4], [1, 2, 8], [0, 7, 5]]
Matriz: [[], [], []]
Seleccione una operación:
```

Sin embargo para terminar de añadir elementos se pulsa *enter* con una fila vacía.

Podemos introducir matrices de diferentes dimensiones, ya sean 3×3 , 2×4 , e incluso 1×1 , aunque algunas operaciones solo funcionan con matrices cuadradas 2×2 y 3×3 (inversa y determinante). Y para el resto tienen que tener sentido matemático, en el producto m_1 tiene que tener el mismo número de filas que m_2 de columnas, esto lo hemos tenido en cuenta devolviendo errores en el servidor.

La implementación de todas las operaciones está en el archivo `servidor.py`.

4. Pruebas y ejemplos:

Vamos a mostrar unos ejemplos de ejecución:

- Operaciones básicas

```
o : Para volver
Ingrese su selección: +
Introduzca el primer número: 5
Introduzca el segundo número: 9
~~ El resultado de la operación + es: 14.0 ~~
*****
v : Para una conversión (grados/radianes)
o : Para volver
Ingrese su selección: c
Introduzca los grados: 90
~~ El resultado de la operación c es: 0.0 ~~
*****
```

```

o : Para volver
Ingrese su selección: t
NO introducir valores indefinidos como 90, 270
Introduzca los grados: 150
~~ El resultado de la operación t es: -0.57735 ~
*****

```

```

t : Para tangente (grados)
v : Para una conversion (grados-radianes)
o : Para volver
Ingrese su selección: v
Introduzca el número: 270
Introduzca las unidades del número anterior (rad o grad): grad
~~ El resultado de la operación v es: 4.71238898038469 ~~
*****

```

- Operaciones con vectores

```

m : Para producto mixto
o : Para volver al modo anterior
Ingrese su selección: -
Selecciona dos vectores (1, 2 o 3) separados por espacios: 2 1
~~ El resultado de la operación - es: [-2.0, -2.0, 5.0] ~~
*****
Vector 1: [5.0, 6.0, 4.0]
Vector 2: [3.0, 4.0, 9.0]
Vector 3: []
Seleccione una operación:

```

```

o : Para volver al modo anterior
Ingrese su selección: x
Los vectores deben tener la misma longitud y ser de dimensión 3
Selecciona dos vectores (1, 2 o 3) separados por espacios: 2 1
~~ El resultado de la operación x es: [-38.0, 33.0, -2.0] ~~
*****
Vector 1: [5.0, 6.0, 4.0]
Vector 2: [3.0, 4.0, 9.0]
Vector 3: []

```

```

m : Para producto mixto
o : Para volver al modo anterior
Ingrese su selección: n
Selecciona un vector (1, 2 o 3): 2
~~ El resultado de la operación n es: 10.295630140987 ~~
*****
Vector 1: [5.0, 6.0, 4.0]
Vector 2: [3.0, 4.0, 9.0]
Vector 3: []

```

```

m : Para producto mixto
o : Para volver al modo anterior
Ingrese su selección: a
Selecciona dos vectores (1, 2 o 3) separados por espacios: 1 2
~~ El resultado de la operación a es: 33.88459387249749 ~~
*****
Vector 1: [5.0, 6.0, 4.0]
Vector 2: [3.0, 4.0, 9.0]
Vector 3: []

```

- Operaciones con matrices

```

d : Para determinante (m1)
i : Para inversa (m1)
o : Para volver al modo anterior
Ingrese su selección: +
~~ El resultado de la operación + es: [[-3.0, -1.0, 4.0], [0.0, 11.0, 13.0], [8.0, 4.0, -2.0]] ~~
*****
Matriz: [[5, -6, 4], [0, 5, 9], [7, 2, -3]]
Matriz: [[-8, 5, 0], [0, 6, 4], [1, 2, 1]]
Seleccione una operación:
q : Para modificar alguna matriz

d : Para determinante (m1)
i : Para inversa (m1)
o : Para volver al modo anterior
Ingrese su selección: x
~~ El resultado de la operación x es: [[-36.0, -3.0, -20.0], [9.0, 48.0, 29.0], [-59.0, 41.0, 5.0]] ~~
*****
Matriz: [[5, -6, 4], [0, 5, 9], [7, 2, -3]]
Matriz: [[-8, 5, 0], [0, 6, 4], [1, 2, 1]]
Seleccione una operación:
q : Para modificar alguna matriz

i : Para inversa (m1)
o : Para volver al modo anterior
Ingrese su selección: i
~~ El resultado de la operación i es: [[0.048316251830161056, 0.014641288433382138, 0.10834553440702782],
[-0.09224011713030747, 0.0629575402635432, 0.06588579795021962], [0.05124450951683748, 0.07613469985358712,
-0.036603221083455345]] ~~
*****
Matriz: [[5, -6, 4], [0, 5, 9], [7, 2, -3]]
Matriz: [[-8, 5, 0], [0, 6, 4], [1, 2, 1]]

x : Para producto de matrices (m1 x m2)
d : Para determinante (m1)
i : Para inversa (m1)
o : Para volver al modo anterior
Ingrese su selección: d
~~ El resultado de la operación d es: -683.0 ~~
*****
Matriz: [[5, -6, 4], [0, 5, 9], [7, 2, -3]]
Matriz: [[-8, 5, 0], [0, 6, 4], [1, 2, 1]]
Seleccione una operación:
q : Para modificar alguna matriz

```

5. Anexos:

Junto con esta memoria se incluye todo el código fuente.