Memoria Calculadora con Thrift

Primera aproximación

Para esta parte utilizando Thrift decidí relaizar la calculadora en Python3 y utilizando el entorno de desarrollo PyCharm. Al igual que la realizada en RPC Sun esta calculadora dispondrá de las operaciones básicas (sumar, restar, multiplicar y dividir) además de la posibilidad de relizar producto escalar y vectorial de vectores de 2 y 3 dimensiones.

El fichero calculadora.thrift quedaría del siguiente modo:

```
struct vect2D
{
    1: required double x;
    2: required double y;
}

struct vect3D
{
    1: required double x;
    2: required double y;
    3: required double z;
}

service Calculadora
{
    void ping();
    double sumar(1:double num1, 2:double num2),
    double restar(1:double num1, 2:double num2),
    double multiplicar(1:double num1, 2:double num2),
    double dividir(1:double num1, 2:double num2),
    double escalar2d(1:vect2D v1, 2:vect2D v2),
    double vectorial2d(1:vect2D v1, 2:vect2D v2),
    double escalar3d(1:vect3D v1, 2:vect3D v2),
    vect3D vectorial3d(1:vect3D v1, 2:vect3D v2);
}
```

En el podemos ver la declaración de los struct para los vectores y luego el servicio Calculadora con los métodos de los que dispondrá.

Tras generar el código con thrift -gen py calculadora.thrift creé un archivo cliente y otro servidor que incluí dentro de la carpeta gen-py y me fui inspirando en los archivos de ejemplo para implementarlos.

La implementación del servidor fue bastante sencilla ya que tan sólo había que realizar las operaciones matemáticas.

En el cliente muestro al usuario las distintas operaciones de las que dispone la calculadora para que este elija una de ellas y el cliente le pedirá los datos necesarios para realizarlas.

También en las clases de los vectores redefiní el método __str__ para así poder mostrar por pantalla correctamente tanto los vectores de 2 como de 3 dimensiones.

Ampliación

Para ampliar aún más esta claculadora decidí crear el servidor en otro lenguaje distinto al utilizado en el cliente. El lenguaje elegido fue Ruby.

En la documentación de Apache Thrift explican cómo implementar un servidor en Ruby y tampoco fue demasiado complicado hacerlo. Lo que más me costó fue el incluir las librerías de Thrift para Ruby ya que en Python había sido tan sencillo como utilizar su controlador de paquetes pip. Finalmente opté por descargar las librerias de GitHub e incluirla en la carpeta gen-rb.

También redefiní el método to_s de los vectores para que se mostrasen correctamente.

Ahora se puede utilizar tanto con el servidor de Python como con el de Ruby.