UNIVERSIDAD EAFIT

INGENIERÍA DE SISTEMAS

ST0263: Tópicos Avanzados en Telemática, grupo 031, 2012-2, Prof.: Edwin Montoya

Reto No 5 – Diseño e Implementación de una Calculadora Web utilizando tecnología de comunicación a nivel de Web Services como Middleware de invocación remota, Web como cliente liviano y Cliente grueso.

Descripción:

Basado en el taller realizado el 27 de octubre de 2012, en el cual se diseño y comenzó a implementar una aplicación Calculadora Web distribuida, realizar la aplicación completa con las siguientes especificaciones:

1. Simular 5 nodos, donde se ejecutaría la aplicación completa, si bien para efectos de desarrollo y pruebas pueden estar en la misma máquina o nodo, los archivos de configuración hacen que se pueda llevar al ámbito distribuido.

2. Implementar un web service: wscalc1 que implemente las operaciones de sumar y multiplicar, en un proyecto AppServer1. Este web service se podrá acceder vía url en su especificación así: http://server:port/AppServer1/wscalc1?wsdl o versión aproximada de acuerdo al entorno y lenguaje empleado.

3. Implementar un web service: wscalc2 que implemente las operaciones de restar y dividir, en un proyecto AppServer2. Este web service se podrá acceder vía url en su especificación así: http://server:port/AppServer2/wscalc2?wsdl o versión aproximada de acuerdo al entorno y lenguaje empleado.

4. Implementar una aplicación (WebCalc) (en jsp, sevlets, php, rails, aspx, etc), que reciba los datos de una forma web con 3 parametros: parametro1: a, parametro2: b, operación: op. Al recibir estos parámetros, esta aplicación web se debe comunicar con el wscalc1 o wscalc2, de acuerdo sea el caso, ejecutar la operación remota y retornar los resultados al browser de nuevo.

5. La aplicación WebCalc leerá de un archivo de configuración xml (calcserver.xml), la localización de los 2 web services, el archivo tiene este formato:

```
<?xml version="1.0" encoding="UTF-8"?>
<urls>
    <url>http://localhost:8080/ws1calc</url>
    <url>http://localhost:8080/ws2calc</url>
</urls>
```

se deberá implementar minimo:

calc.html (forma de entrada de datos)

calcservlet.java, calc.jsp, calc.php, o calc.* de acuerdo a la tecnología (aplicación web que procesa los datos del browser, lee el archivo de configuración, invoca los web services remotos, y arma la respuesta html al browser)

en el caso del diseño con java (puede ser diferente para otros entornos), se puede crear otro archivo proxy para ir a los web services remotos (calcproxy.java), el cual es llamado por el proxy. En este calcproxy.java, realizará la instanciación de los clientes de los web services, mas o menos así:

wscalc1_service cs = new wscal1_service(new URL(string_url_wscalc1));

wscalc1 calc1 = cs.getWscalc1();

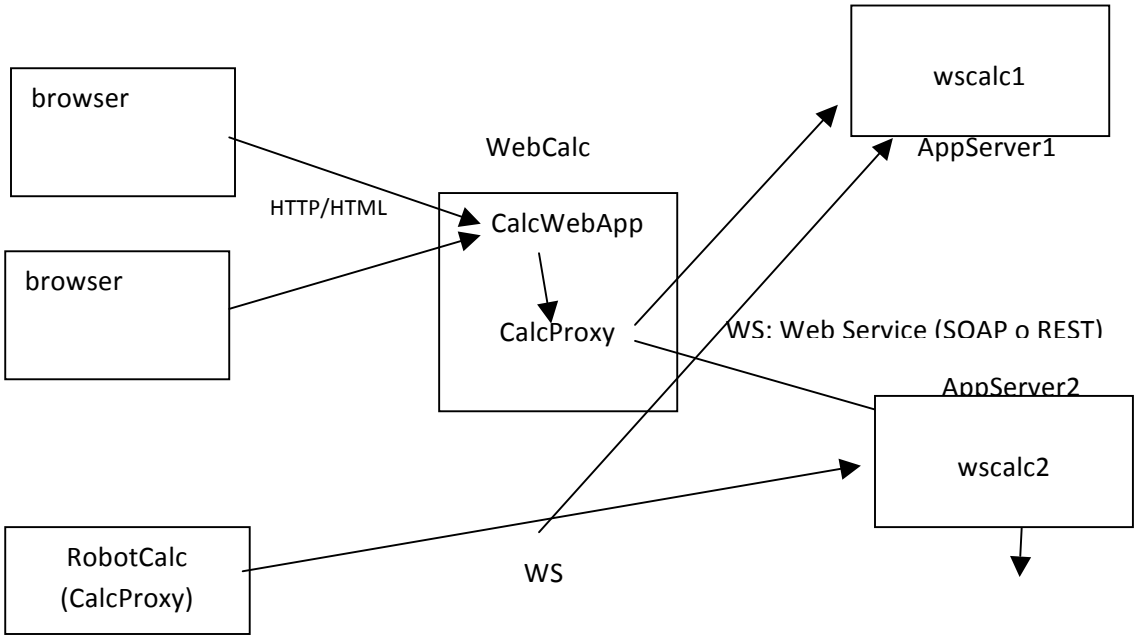
// y ya podrá llamar los métodos transparentes de calc1.

6. Realizar una aplicación standalone o cliente grueso (RobotCalc), que de acuerdo a un archivo de entrada calc.xml, invoque los web services remotos, de acuerdo a los métodos y parámetros del archivo, y escriba la respuesta en el tag <result>.

   Este RobotCalc, leerá tanto el archivo calcserver.xml para localizar los web services remotos, como el archivo calc.xml donde están las operaciones a ejecutar.

   El formato del archivo y un ejemplo del calc.xml es así:

```
<?xml version="1.0" encoding="UTF-8"?>
<webservices>
  <webservice>
    <url>http://server1:port/AppServer1/wscalc1</url>
    <method>sumar</method>
    <params>
      <p1>10</p1>
      <p2>20</p2>
    </params>
    <result>30</result>
  </webservice>
  <webservice>
    <url>http://server2:port/AppServer2/wscalc2</url>
    <method>restar</method>
    <params>
      <p1>10</p1>
      <p2>20</p2>
    </params>
    <result>-10</result>
  </webservice>
</webservices>
```



Entregables:

- Archivos fuentes del reto
- Documentación básica: Integrantes, diseño, implementación y ejecución del reto en uno de los servidores dispuestos para la práctica.