

CICLE FORMATIU: CFGS Desenvolupament Aplicacions Multiplataforma DAM
MÒDUL PROFESSIONAL: MP09 Programació de serveis i processos
UNITAT FORMATIVA: UF3. Sòcols i serveis
ACTIVITAT: Invocació de mètodes remots

=====

Pràctica 3. Invocació de mètodes remots.

En aquesta pràctica crearem una aplicació client/servidor que farà servir mètodes remots del servidor, mitjançant RMI en java.

- Busca al moodle l'enllaç al codi del exemple de la calculadora remota.
- L'objectiu és implementar els mètodes sumar, restar, multiplicar i dividir en el costat del servidor, i poder accedir a ells directament des del programa client.
- Et pot ajudar també el codi de l'enllaç "calculadora simple RMI" del moodle.
- Quan ja et funcionin els quatre mètodes anteriors, afegeix un nou mètode "potencia" que calculi un número elevat a un altre.

Pràctica

Entrega un document amb els programes client i servidor, i una breu explicació del que fa cada un d'ells. Codi font comentat. Fes captures d'execucions.

CODIGO CLIENTE

```
package calculadora;  
  
import java.rmi.RemoteException;  
import java.rmi.registry.LocateRegistry;  
import java.rmi.registry.Registry;  
  
public class ClienteTCP_Calculadora {
```

```

    public static void main(String[] args) throws RemoteException {

        RMICalcInterface calc = null;
        try {
            System.out.println("Localizando registro de objetos remotos...");
            Registry registry = LocateRegistry.getRegistry("192.168.41.219",
5555); // CREAMOS UN REGISTRO HACIA LA IP DEL SERVIDOR EN EL
PUERTO 5555
            System.out.println("Obteniendo el stub del objeto remoto...");
            calc = (RMICalcInterface) registry.lookup("Calculadora"); //
AQUI ESTA LEYENDO EL REGISTRO CREADO EN EL SERVIDOR
"CALCULADORA"
        } catch (Exception e) {
            e.printStackTrace();
        }

        if (calc != null) {
            System.out.println("Realizando operaciones con el objeto
remoto...");

            try {
                System.out.println("Sumando 2 y 2 : " + calc.suma(2, 2));
                System.out.println("Restando 99 y 45 : " + calc.resta(99,
45));
                System.out.println("Multiplicando 125 por : 3 " +
calc.multip(125, 3));
                System.out.println("Diviando 10 entre 5 : " + calc.div(10,
5));
                System.out.println("Elevando 2 a 3 : " + calc.potencia(2,
3));
            } catch (Exception e) {
                e.printStackTrace();
            }
            System.out.println("Terminado");
        }
    }
}

```

CODIGO SERVIDOR

```

package calculadora;

import java.rmi.RemoteException;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.UnicastRemoteObject;

public class ServidorTCP_Calculadora implements RMICalcInterface { //

```

IMPLEMENTO LA INTERFACE CREADA ANTERIORMENTE

// DEFINIR LOS METODOS AL IMPLEMENTAR LA INTERFACE

```
public int suma(int a, int b) throws RemoteException {
    System.out.println("Sumando " + a + " y " + b + "...");
    return (a + b);
}

public int resta(int a, int b) throws RemoteException {
    System.out.println("Restando " + a + " y " + b + "...");
    return (a - b);
}

public int multip(int a, int b) throws RemoteException {
    System.out.println("Multiplicando " + a + " por " + b + "...");
    return (a * b);
}
```

```
public int div(int a, int b) throws RemoteException {
    System.out.println("Dividiendo " + a + " entre " + b + "...");
    return (a / b);
}
```

@Override

```
public int potencia(int a, int b) throws RemoteException {
    System.out.println("Elevando " + a + " a la potencia " + b + "...");
    return (int) Math.pow(a, b);
}
```

```
public static void main(String[] args) {
```

// CREAMOS UN REGISTRO DE OBJETOS REMOTOS

```
System.out.println("Creando el registro de objetos remotos...");
Registry reg = null;
```

// ABRIMOS EL REGISTRO EN EL PUERTO 5555

```
try {
    reg = LocateRegistry.createRegistry(5555);
} catch (Exception e) {
    System.out.println("Error: No se ha podido crear el registro");
    e.printStackTrace();
}
```

//CREAMOS EL OBJETO SERVIDOR Y LO INSCRIBIMOS EN EL REGISTRO.

```
System.out.println("Creando el objeto servidor e inscribiendolo en el
registro...");
ServidorTCP_Calculadora serverObject = new
```

```
ServidorTCP_Calculadora();
```

//FINALMENTE LE DAMOS UN NOMBRE AL REGISTRO "CALCULADORA" POR EL CUAL EL CLIENTE PODRA ENTRAR Y RESOLVER SUS OPERACIONES.

```
        try {  
            reg.rebind("Calculadora", (RMICalcInterface)  
UnicastRemoteObject.exportObject(serverObject, 0));  
        } catch (Exception e) {  
            System.out.println("Error: No se ha podido inscribir el objeto  
servidor");  
            e.printStackTrace();  
        }  
    }  
}
```

CLASE INTERFACE

```
package calculadora;
```

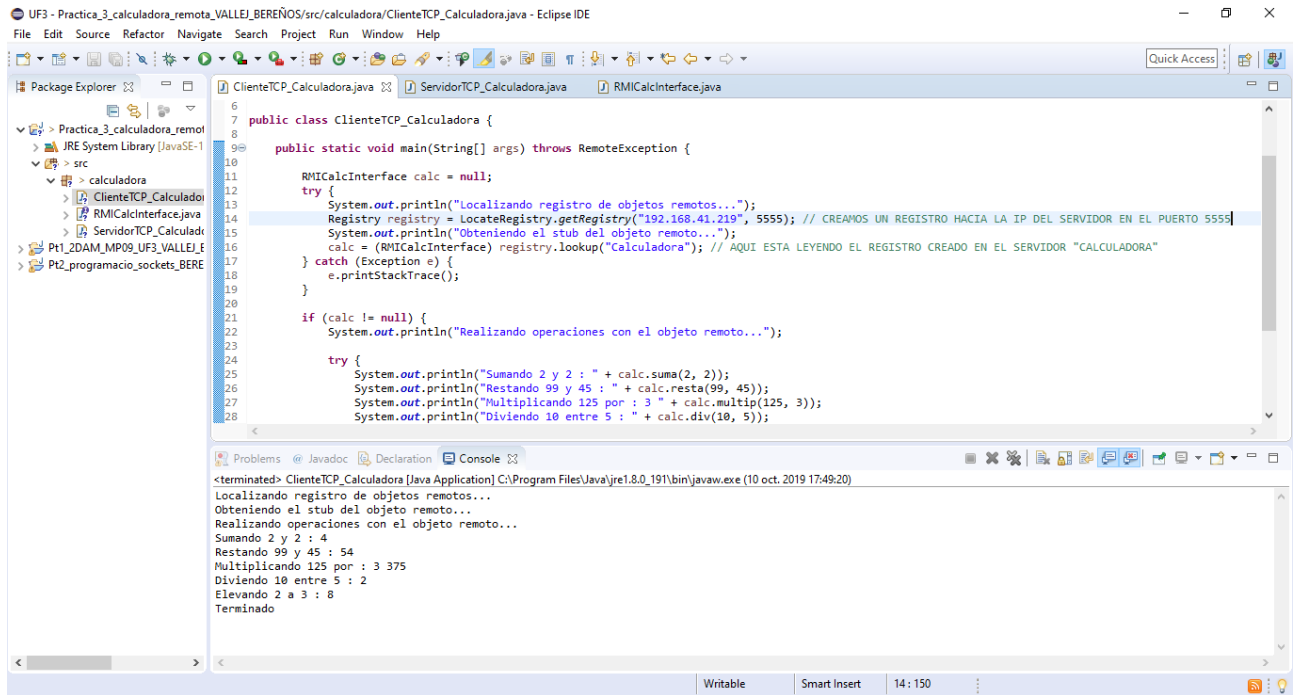
```
import java.rmi.Remote;  
import java.rmi.RemoteException;
```

```
public interface RMICalcInterface extends Remote{
```

// CREO UNA INTERFACE PARA DECLARAR LOS METODOS DE LAS DIFERENTES OPERACIONES DE LAS CALCULADORAS

```
    public int suma(int a, int b) throws RemoteException;  
  
    public int resta(int a, int b) throws RemoteException;  
  
    public int multip(int a, int b) throws RemoteException;  
  
    public int div(int a, int b) throws RemoteException;  
  
    public int potencia(int a, int b) throws RemoteException;  
  
}
```

FOTO RESULTADO LADO DEL CLIENTE



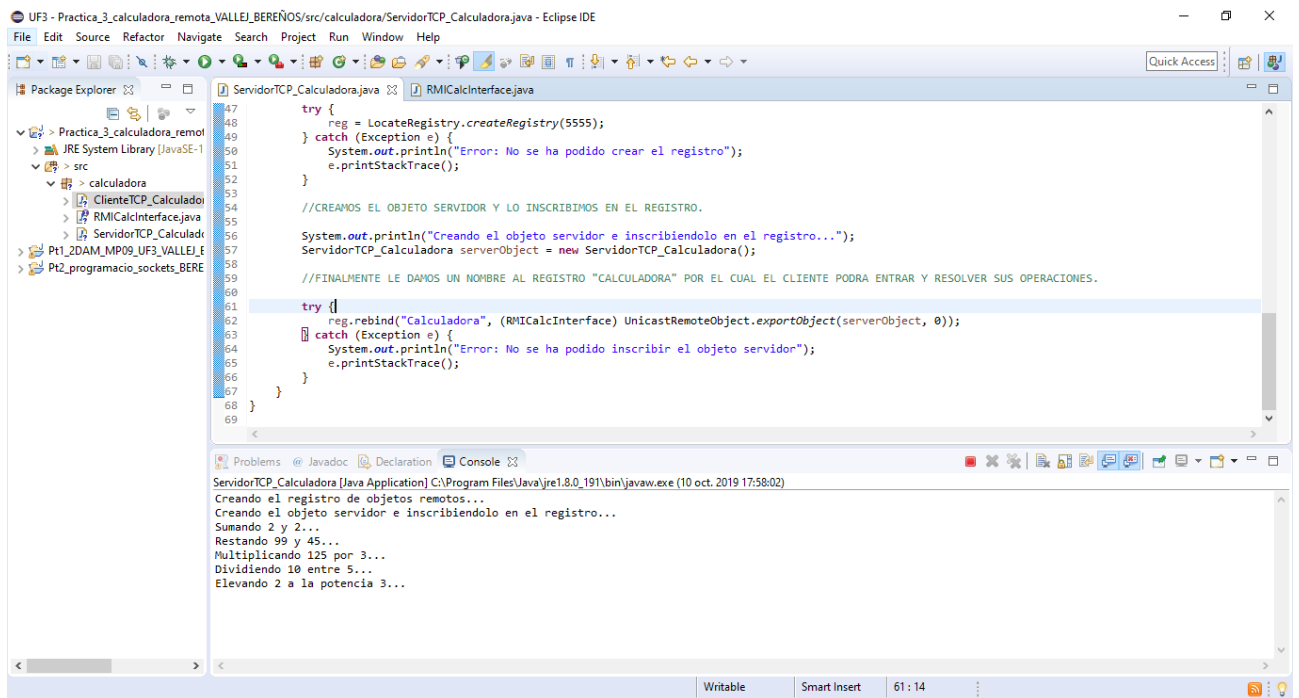
The screenshot shows the Eclipse IDE with the file `ClienteTCP_Calculadora.java` open. The code defines a `ClienteTCP_Calculadora` class with a `main` method that performs remote calculations using RMI. The console output shows the successful execution of these operations.

```
public class ClienteTCP_Calculadora {  
    public static void main(String[] args) throws RemoteException {  
        RMICalcInterface calc = null;  
        try {  
            System.out.println("Localizando registro de objetos remotos...");  
            Registry registry = LocateRegistry.getRegistry("192.168.41.219", 5555); // CREAMOS UN REGISTRO HACIA LA IP DEL SERVIDOR EN EL PUERTO 5555  
            System.out.println("Obteniendo el stub del objeto remoto...");  
            calc = (RMICalcInterface) registry.lookup("Calculadora"); // AQUI ESTA LEYENDO EL REGISTRO CREADO EN EL SERVIDOR "CALCULADORA"  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
        if (calc != null) {  
            System.out.println("Realizando operaciones con el objeto remoto...");  
            try {  
                System.out.println("Sumando 2 y 2 : " + calc.suma(2, 2));  
                System.out.println("Restando 99 y 45 : " + calc.resta(99, 45));  
                System.out.println("Multiplicando 125 por : 3 " + calc.multip(125, 3));  
                System.out.println("Dividiendo 10 entre 5 : " + calc.div(10, 5));  
            }  
        }  
    }  
}
```

Console Output:

```
<terminated> ClienteTCP_Calculadora [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (10 oct. 2019 17:49:20)  
Localizando registro de objetos remotos...  
Obteniendo el stub del objeto remoto...  
Realizando operaciones con el objeto remoto...  
Sumando 2 y 2 : 4  
Restando 99 y 45 : 54  
Multiplicando 125 por : 3 375  
Dividiendo 10 entre 5 : 2  
Elevando 2 a 3 : 8  
Terminado
```

FOTO RESULTADO LADO SERVIDOR



The screenshot shows the Eclipse IDE with the file `ServidorTCP_Calculadora.java` open. The code defines a `ServidorTCP_Calculadora` class that implements the `RMICalcInterface` and registers itself with the RMI registry. The console output shows the successful registration and execution of the server.

```
try {  
    reg = LocateRegistry.createRegistry(5555);  
} catch (Exception e) {  
    System.out.println("Error: No se ha podido crear el registro");  
    e.printStackTrace();  
}  
//CREAMOS EL OBJETO SERVIDOR Y LO INSCRIBIMOS EN EL REGISTRO.  
System.out.println("Creando el objeto servidor e inscribiendolo en el registro...");  
ServidorTCP_Calculadora serverObject = new ServidorTCP_Calculadora();  
//FINALMENTE LE DAMOS UN NOMBRE AL REGISTRO "CALCULADORA" POR EL CUAL EL CLIENTE PODRA ENTRAR Y RESOLVER SUS OPERACIONES.  
try {  
    reg.rebind("Calculadora", (RMICalcInterface) UnicastRemoteObject.exportObject(serverObject, 0));  
} catch (Exception e) {  
    System.out.println("Error: No se ha podido inscribir el objeto servidor");  
    e.printStackTrace();  
}
```

Console Output:

```
ServidorTCP_Calculadora [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (10 oct. 2019 17:58:02)  
Creando el registro de objetos remotos...  
Creando el objeto servidor e inscribiendolo en el registro...  
Sumando 2 y 2...  
Restando 99 y 45...  
Multiplicando 125 por 3...  
Dividiendo 10 entre 5...  
Elevando 2 a la potencia 3...
```