

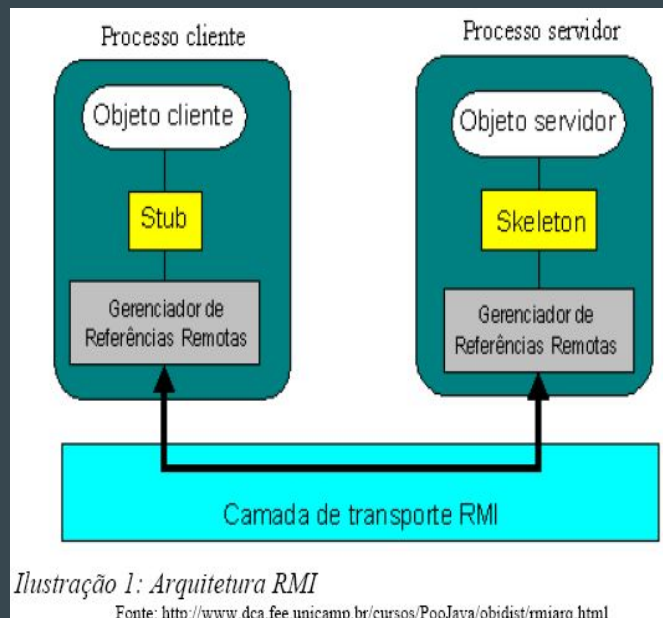
Java RMI

...

Arthur Haas Dorneles

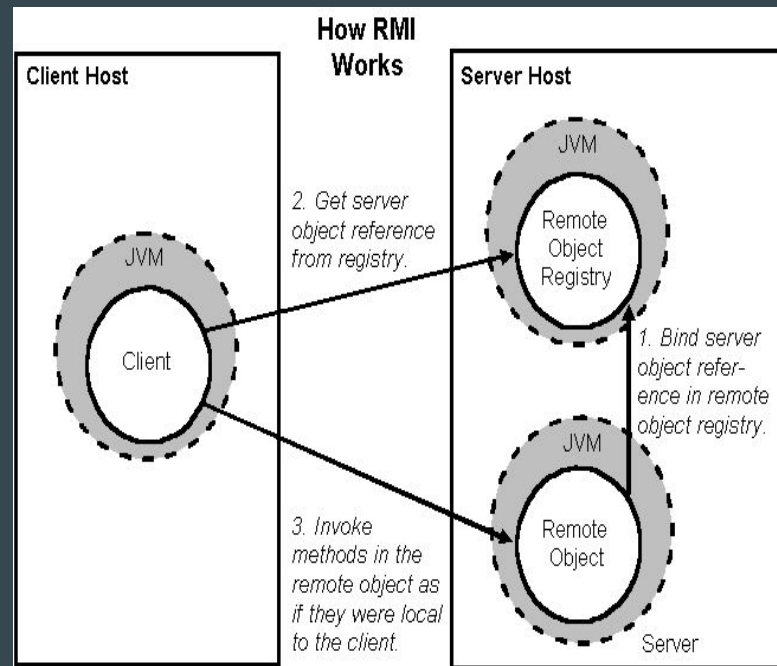
Java RMI

- RMI (Remote Method Invocation).
- Dividido em 3 camadas:
 - Stub/Skeleton
 - Referências Remotas
 - Transporte



Como funciona?

- Stub e Skeletons (Marshalling)
- RRL (Locate Registry)
- Transporte
 - Endpoint
 - Transporte
 - Canal
 - Conexão



Implementação

- Calculadora que pode realizar somas, subtrações, multiplicações e divisões
- Interface da Calculadora:

```
1+ import java.rmi.Remote;
3
4 public interface Calculadora extends Remote{
5     public double calc(double a, double b, char c) throws RemoteException;
6 }
```

Implementação

- Implementação da interface Calculadora:

```
1 import java.rmi.RemoteException;
2
3 public class CalculadoraImple extends UnicastRemoteObject implements Calculadora{
4
5     private static final long serialVersionUID = 1L;
6
7     protected CalculadoraImple() throws RemoteException{
8         super();
9     }
10
11     public double calc(double a, double b, char op) throws RemoteException{
12         double i = 0;
13         if(op == '+') {
14             i = this.add(a, b);
15         }else if(op == '-') {
16             i = this.sub(a, b);
17         }else if(op == '*') {
18             i = this.mult(a, b);
19         }else if(op == '/') {
20             i = this.div(a, b);
21         }
22         return i;
23     }
24
25     private double add(double a, double b){
26         System.out.println("A = " + a + " B = " + b);
27         return a+b;
28     }
29
30     private double sub(double a, double b){
31         System.out.println("A = " + a + " B = " + b);
32         return a-b;
33     }
34
35     private double mult(double a, double b){
36         System.out.println("A = " + a + " B = " + b);
37         return a*b;
38     }
39
40     private double div(double a, double b){
41         System.out.println("A = " + a + " B = " + b);
42         return a/b;
43     }
44 }
```

Implementação

- Servidor:

```
1 import java.rmi.Naming;
2
3
4
5 public class CalculadoraServer {
6     CalculadoraServer() {
7         try {
8             System.setProperty("java.rmi.server.hostname", "192.168.0.15");
9             LocateRegistry.createRegistry(1099);
10            Calculadora c = new CalculadoraImple();
11            Naming.bind("CalculadoraService", (Remote) c);
12        } catch (Exception e) {
13            e.printStackTrace();
14        }
15    }
16
17    public static void main(String[] args) {
18        new CalculadoraServer();
19    }
20 }
21
```

Implementação

- Cliente:

```
1 import java.rmi.Naming;
2
3 public class CalculadoraClient {
4     public static void main(String[] args) {
5         try {
6             Calculadora c = (Calculadora) Naming.lookup("rmi://192.168.0.15:1099/CalculadoraService");
7             Scanner l = new Scanner(System.in);
8             double a,b;
9             char op;
10            int d = 0;
11            while(d == 0){
12                System.out.println("0-Calculadora\n1-Sair");
13                d = l.nextInt();
14                if(d == 0){
15                    System.out.println("Informe os valores a serem calculados e a operação na seguinte forma -> <Valor1> <Operação> <Valor2>:");
16                    System.out.println("Valor 1");
17                    a=l.nextDouble();
18                    System.out.println("Operador");
19                    op = (char)System.in.read();
20                    System.out.println("Valor 2");
21                    b=l.nextDouble();
22                    System.out.println("Resultado : " + c.calc(a,b,op));
23                }
24            }
25            l.close();
26        } catch (Exception e) {
27            e.printStackTrace();
28        }
29    }
30 }
31
```

Execução

- Compilar (javac)
- Gerar stub (rmic)
- Executar (java)
- github.com/DornelesArthur/RMI