

Exercício 1 Sistemas Distribuídos

Christian Aguiar Plentz¹, João Roberto Lemos Fidellis²

¹Ciência da Computação – Universidade do Vale do Rio dos Sinos (UNISINOS)
Caixa Postal 275 – 93022-750 – São Leopoldo – RS – Brazil

{plentzchristian, JRFIDELLIS}@edu.unisinos.br

1. Exercícios

A seguir são os exercícios da aula 3 de Sistemas Distribuídos do Grau A

1.1. Por que sistemas em rede são organizados em camadas?

Cada camada da Internet provém utilidade para a camada acima usufruir com a abstração da camada subsequente. Então um programador não precisa se preocupar com a responsabilidade da integridade dos dados mandados se ele usar a tecnologia TCP/IP que já mantém essa parte de um programa.

1.2. Quais as vantagens da arquitetura ”Computadores em Rede”?

A vantagem é que dependendo da implementação de responsabilidade de cada computador nessa rede se um Computador em uma rede cai, todo o serviço não cai junto com ele.

1.3. Porque a arquitetura cliente-servidor tem problemas de escalabilidade?

Porque, à medida que o número de clientes cresce, o servidor central vira um ponto único de sobrecarga: ele precisa processar todas as requisições, o que pode gerar lentidão, gargalos e até falhas se não houver recursos suficientes.

1.4. Por que replicação funciona bem quando a maioria das operações forem de leitura!

Por que operações de escrita tem a complexidade relacionada a propagar os dados em todas as réplicas. Se escrever em uma das réplicas, preciso no mínimo invalidar o dado nas demais réplicas ou substituir com a nova versão da informação.

1.5. O que você entende pelo processo de reflexão (Java) (recursivo)

O processo de reflexão consiste no processo consultando informações sobre classes e objetos em tempo de execução. Exemplo: Durante execução, chamar um método de `java.lang.reflect` para obter uma lista dos métodos de uma classe

2. Implementacao RMI Java

2.1. Código RMI, execute e comente o código e a execução. Pode ser qualquer código.

Listing 1. Classe CalcImpl e Calc

```
package calcRmi;
public interface Calc extends Remote {
    int add(int i, int j) throws RemoteException;
}
public class CalcImpl extends UnicastRemoteObject
    implements Calc {
    public CalcImpl() throws RemoteException {
        super();
    }

    public int add(int i, int j) {
        return i + j;
    }
}
```

Listing 2. Cliente

```
package client;
public class Main {
    public static void main(String[] args) {
        try {
            Calc objCalc = (Calc)Naming.lookup("//localhost
                :2004/Calc");
            //Busca o ip localhost na porta 2004 pelo Objeto Calc
            System.out.println("O resultado da soma é: " +
                objCalc.add(3, 7));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Listing 3. Servidor

```
package server;
class Main {
    public static void main(String[] args){
        try {
            //Usa a porta 2004 para receber chamadas
            LocateRegistry.createRegistry(2004);
            CalcImpl objRMI = new CalcImpl();
            //Ouve o IP localhost porta 2004 com a classe Calc
            Naming.rebind("//127.0.0.1:2004/Calc", objRMI);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
}  
}
```

O servidor não imprime nada mas quando o servidor esta rodando e o cliente for rodado o cliente tera este Output no terminal.

```
$ java client.Main  
O resultado da soma é : 10
```

Se o servidor não estiver rodando ele vai lançar uma exceção de que a conexão foi rejeitada

```
$ java client.Main  
java.rmi.ConnectException: Connection refused to host:  
→ localhost; nested exception is:  
    java.net.ConnectException: Connection refused  
    at java.rmi/sun.rmi.transport.tcp.TCPEndpoint.newSocket(TCPEndpoint.java:625)  
    at java.rmi/sun.rmi.transport.tcp.TCPChannel.createConnection(TCPChannel.java:217)  
    at java.rmi/sun.rmi.transport.tcp.TCPChannel.newConnection(TCPChannel.java:204)  
    at java.rmi/sun.rmi.server.UnicastRef.newCall(UnicastRef.java:345)  
    at java.rmi/sun.rmi.registry.RegistryImpl_Stub.lookup(RegistryImpl_Stub.java:116)  
    at java.rmi/java.rmi.Naming.lookup(Naming.java:101)  
    at client.Main.main(Main.java:9)  
Caused by: java.net.ConnectException: Connection refused  
    at java.base/sun.nio.ch.Net.connect0(Native Method)  
    at java.base/sun.nio.ch.Net.connect(Net.java:589)  
    at java.base/sun.nio.ch.Net.connect(Net.java:578)  
    at java.base/sun.nio.ch.NioSocketImpl.connect(NioSocketImpl.java:583)  
    at java.base/java.net.SocksSocketImpl.connect(SocketImpl.java:327)  
    at java.base/java.net.Socket.connect(Socket.java:751)  
    at java.base/java.net.Socket.connect(Socket.java:686)  
    at java.base/java.net.Socket.<init>(Socket.java:555)  
    at java.base/java.net.Socket.<init>(Socket.java:324)  
    at java.rmi/sun.rmi.transport.tcp.TCPDirectSocketFactory.createSocket(TCPDirectSocketFactory.java:40)
```

```
at java.rmi/sun.rmi.transport.tcp.TCPEndpoint.newSocket(TCPEndpoint.java:619)
... 6 more
```