Comunicação entre Processos com RPC/RMI

CK0154 – Sistemas Distribuídos - Trabalho III

Javam Machado - Eduardo Rodrigues - Jose Serafim Filho

1 Introdução

A comunicação entre processos (IPC) é um mecanismo que permite o compartilhamento de dados entre processos. Esse mecanismo possibilita o desenvolvimento de aplicações distribuídas, ou seja, aplicações que dividem o processamento entre computadores através de uma rede. Um dos métodos de implementação de IPC se dá por troca de mensagens utilizando-se por exemplo RPC/RMI.

2 Especificação

2.1 Objetivo

Implementar duas aplicação distribuídas distintas, A primeira será uma calculadora e a segunda um transmissor de mensagens para um grupo de clientes. Essas aplicações deverão ser suportadas por uma arquitetura de cliente/servidor em que a troca de mensagens seja feita por RPC ou RMI como mostra a Figura 1.

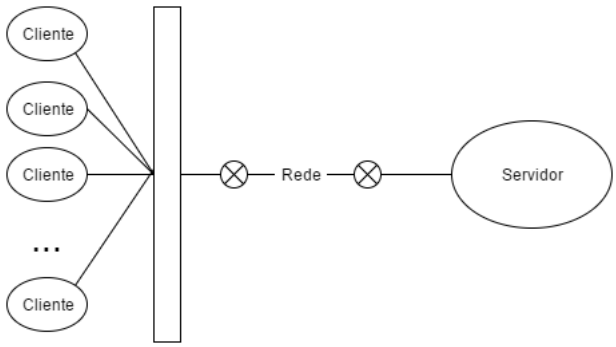

Figure 1: Aplicação distribuída

2.2 Resumo das atividades para calculadora

- Implementar servidor capaz de:
  - Calcular operações +, -, *, /;
- Implementar cliente capaz de:
  - Submeter dois números e uma operação matemática: +, -, *, /;
  - Visualizar resultado da operação

  Esclarecimentos:
  - Faça um menu simples no cliente para a chamada das operações, deixando claro como deve ser feita a entrada de dados pelos usuários;

2.3 Resumo das atividades para troca de mensagem

- Implementar servidor capaz de:
  - Manter uma lista de clientes conectados;
  - Receber mensagem de texto de um cliente e encaminhá-la para todos os outros clientes (não enviar a mensagem para o cliente que enviou);
- Implementar cliente capaz de:
  - Solicitar participação na troca de mensagens;
  - Solicitar saída da troca de mensagens;
  - Enviar mensagem de texto (uma palavra) para o servidor;
  - Visualizar mensagens recebidas

Esclarecimentos:

- O estado inicial do sistema contém 1 servidor (possui uma lista de clientes vazia) e quantos clientes forem abertos;
- Ao solicitar participação na troca de mensagens, o cliente é adicionado a lista, e ao solicitar a saída o cliente é removido da mesma;
- O servidor deve ser capaz de identificar o cliente que enviou a mensagem de texto a ser retransmitida e enviar a todos os outros clientes presentes na lista sem re-enviar ao cliente emissor;
- Os clientes e o servidor deverão ficar em loop, não encerrando sua execução depois de uma operação. Façam um menu simples (não precisa de interface gráfica) para selecionar as operações dos clientes;

Dicas:

- Tanto os clientes, como o servidor deverão fazer o bind com o stub, usando um nome específico que será usado na operação de lookup, ex:
  Servidor - registry.bind("Service", stub)
  Cliente1 - registry.bind("Cliente1", stub)
  Cliente2 - registry.bind("Cliente2", stub)

- Os clientes ao solicitarem a participação na troca de mensagens, deverão enviar ao servidor a string usada na operação de bind, pois esta será usada pelo servidor na hora do lookup para enviar a mensagem a todos os clientes. ...
  O que permitirá que tanto o cliente como o servidor envie mensagens de send/request, não apenas mensagens de receive/reply.

2.4 Critério de avaliação

- Implementar aplicação da calculadora distribuída usando RMI ou RPC(50%).
- Implementar aplicação de troca de mensagens distribuída usando RMI ou RPC(50%).

2.5 Equipe

- Trabalho em dupla;

2.6 Linguagens de programação e API

O trabalho poderá ser realizado nas seguintes linguagens de programação:

- C++
- C
- Java
- Python

2.7 Entrega

O trabalho deverá ser enviado até 23:59 do dia 24/09/17 (domingo) ao Bitbucket. O aluno deve criar uma nova pasta("Trabalho3") no seu repositório no BitBucket. Nessa pasta deve conter o código fonte(incluindo comentários) e um arquivo de texto ou pdf especificando os autores.

References

[1] https://docs.oracle.com/javase/8/docs/technotes/guides/rmi/hello/hello-world.html

[2] http://rpclib.net/

[3] https://github.com/facebook/wangle/blob/master/wangle/example/rpc/RpcClient.cpp