Aluno: Ícaro Pires de Souza Aragão **Matrícula:** 15/0129815
Disciplina: Programação para Sistemas Paralelos e Distribuídos
Professor: Fernando William Cruz

Remote Method Invocation

1. Introdução

Semelhante à tecnologia RPC (Remote Procedure Call), temos o RMI (Remote Method Invocation). A principal diferença entre elas é que na segunda, ao invés de fazermos chamadas remotas de funções pela rede, são chamados métodos de objetos java, ou seja, um objeto é vinculado a um endereço e então eventuais clientes podem fazer uso dele.

Baseado nisso, foi proposto que fosse feita uma pequena aplicação usando-se desta tecnologia, um cliente, que pode estar em um host separado do servidor, usaria um objeto remoto disponibilizado pelo servidor para fazer uma operação matemática simples (adição, subtração, multiplicação ou divisão).

Além disso, também foi proposto que se fizesse numa tecnologia com o mesmo princípio porém na linguagem Ruby e que fosse verificado a disponibilidade de que uma das partes (cliente ou servidor) se comunicasse com a outra numa linguagem diferente.

2. Solução

2.1. Java

Na solução adotada foi utilizada a estrutura básica para uma comunicação RMI, conforme indicado na documentação oficial. Foram implementados:

- Uma interface que herda de Remote, no próprio módulo rmi do java, ela também irá exigir que o objeto que será disponbilizado terá que implementar as operações básicas matemáticas e um método para inicializar os operandos; ***(Operation.java)***
- Um servidor, que vincula o objeto compartilhado a um host e porta; ***(OperationServer.java)***
- Um cliente, que irá acessar o objeto disponibilizado, coletar os operandos e operador do usuário, realizará a operação e imprimira o resultado; ***(OperationClient.java)***
- Uma classe cujos objetos poderão ser compartilhados, pois ele implementa a interface Operation.java e portanto atende aos requisitos definidos; ***(example_Operation.java)***
- Um pequeno script para compilar e gerar os arquivos da política de segurança. ***(make)***

2.2. Ruby

A solução em ruby foi utilizando a biblioteca padrão *drb* (distributed ruby) e foi bem mais simples com relação a solução em java, exigiu apenas dois arquivos:

- O servidor, que implementa a classe do objeto que irá ser compartilhado, instancia o objeto e o vincula na rede ***(server.rb)***;
- O cliente coleta as entradas do usuário atráves de interação em tempo de execução usa o objeto compartilhado para realizar a operação matemática e imprime o resultado ***(client.rb)****;*

3. Comparativo

Ambas soluções utilizaram o mesmo princípio, porém possuem implementações bem específicas. Devido a isso, utilizando a biblioteca *rmi* do java e a biblioteca *drb* do ruby, não é possível

que partes de programas se comuniquem entre si, para isso seria necessário uma outra implementação, por exemplo a CORBA, que teria suporte para fazer as chamadas remotas de método, porém, entre linguagens diferentes.

No aspecto da implementação, a solução em ruby foi bem mais compacta, afinal, a linguagem java geralmente exige operações mais extensas por natureza. Por outro lado, a solução em java é mais segura, pois exigiu que alguns aspectos de segurança fossem adicionados para que houvesse comunicação entre *hosts* distintos.

4. Aspectos Gerais

Com esse experimento é possível reforçar os conceitos de RPC e observar como eles puderam ser aplicados a tecnologias mais modernas, além de observar o funcionamento de diferentes implementações.

5. Dificuldades

As maiores dificuldades ocorreram ao tentar executar a aplicação em java em *hosts* distintos. Os problemas ocorreram devido a duas questões.

A primeira diz respeito a aspectos de segurança, era necessário instanciar um gerenciador de segurança, caso já não houvesse, e também adicionar políticas de segurança ao servidor e ao cliente, a aplicação não funciona caso a política não seja especificada.

O segundo problema foi justamente por causa da facilitação do java, porque ele abstrai algumas etapas que seriam manuais no caso de estar utilizando outra biblioteca de uma outra linguagem. A facilitação que estava atrapalhando era a de que os mecanismos do java já escolhem automaticamente o *host* em que o objeto será vinculado, porém ele estava sempre vinculando o objeto no endereço *localhost*. Para resolver isso foi necessário alterar o arquivo em /*etc/hosts*, mapeando o *hostname* que antes apontava para o IP 127.0.1.1 para o que foi atribuído pelsa interface de rede.