Christian Reis Costa
Egenharia de Software

Comunicação Remota – Introdução

Socket

O acesso ao serviço da camada de transporte pode ser feita por primitivas de transporte, essas primitivas são denominadas SOCKET. Por essas primitivas é possível acessar vários protocolos da camada de transporte, dentre elas TCP e UDP.

Sockets garantem a intercomunicção bidirecional entre processos, executados localmente ou máquinas conectadas através de uma LAN/WAN. Sockets criados usam endereços para fazer referência entre si. O espaço de possíveis endereços é denominado domínio. No caso de sockets alguns domínios possíveis são:

- AF_UNIX – endereço é composto por caminho dentro do sistema de arquivos. O domínios está restrito à arvore de diretórios acessível pelo processo que criou o socket. Utilizado quando os processos rodam em uma mesma máquina.
- AF_INET – o endereço é composto pelo endereço de rede da máquina (IP) e o número de identificação da porta sendo utilizada pelo processo.

Tipos de socket:

- SOCK_STREAM – indica que os dados irão trafegar pelo socket na forma de um stream de caracteres.
- SOCK_DGRAM – indica que os dados irão trafegar na forma de datagramas.

Primitivas – Funções

Socket

Cria um novo ponto final de comunicação e aloca espaço ma entidade transporte local para tratamento da conexão. Não atribui diretamente um endereço ao socket criado.

Bind

Anexa um endereço local a um socket. Um processo servidor deve executar essa primitiva para disponibilizar um endereço aos clientes. Na arquitetura cliente/servidor após sua execução os clientes podem se conectar ao servidor.

Listen

Torna o processo servidor apto a aceitar conexões dos clientes, alocam uma fila de espera para conexões pendentes. Se a fila já foi alocada mostra o tamanho. Caso seja solicitada conexão e a lista de espera estiver cheia é enviada uma mensagem de erro ao solicitante.

Accept

Faz com o processo servidor permaneça bloqueado até que uma solicitação de conexão seja feita.

Connect

Solicita uma conexão ao servidor, bloqueia o processo até que a conexão seja estabelecida. Essa primitiva deve ser executada por um processo cliente para estabelecer comunicação com o servidor.

Send ou Write

Envia dados pela conexão para a entidade destino

Receive ou Read

Recebe dados da conexão

Close

Termina a conexão. Como o encerramento da conexão é simétrico apenas quando ambos os processos tiverem executando a primitiva a conexão será terminada.

RMI

O RMI (Remote Method Invocation) é uma interface de programação que permite a execução de chamadas remotas no estilo RPC em aplicações desenvolvidas em Java. É uma das abordagens da plataforma Java para prover soluções as funcionalidades de uma plataforma de objetos distribuídos. Esse sistema de objetos distribuídos faz parte do núcleo básico de Java desde a versão SDK 1.1, com sua API sendo especificada através do pacote java.rmi e seus subpacotes.

Através da utilização da arquitetura RMI, é possível que u objeto ativo em uma máquina virtual Java possa interagir com objetos em outras máquinas virtuais Java, independentemente da localização dessas máquinas virtuais.

A API RMI fornece ferramentas para que seja possível ao programador desenvolver uma aplicação sem se preocupar com detalhes de comunicação entre diversos possíveis elementos(hosts) de um sistema.

RPC

O RPC (Remote Procedure Call) é uma tecnologia de comunicação entre processos que permite a um programa de computador chamar um procedimento em outro espaço de endereçamento (geralmente em outro computador, conectado por uma rede). O programador

não se preocupa com detalhes de implementação dessa interação remota: do ponto de vista do código, a chamada se assemelha a chamadas de procedimentos locais.

RPC é uma tecnologia popular para a implementação do modelo cliente-servidor de computação distribuída. Uma chamada de procedimento remoto é iniciada pelo cliente enviando uma mensagem para um servidor remoto para executar um procedimento específico. Uma resposta é retornada ao cliente. Uma diferença importante entre chamadas de procedimento locais é que, no primeiro caso, a chamada pode falhar por problemas da rede. Nesse caso, não há nem mesmo garantia de que o procedimento foi invocado.