Sistemas Distribuídos

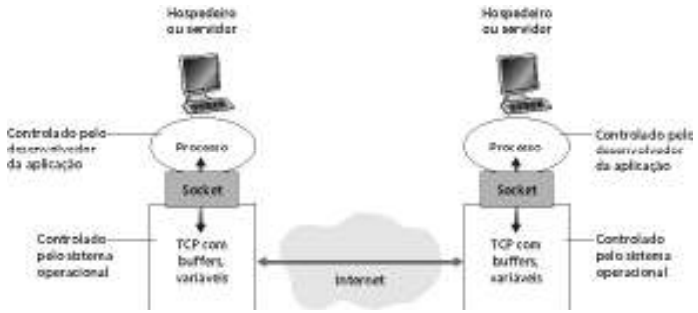
Sockets, RPC, RMI-Java e CORBA

6

Universidade Estácio de Sá
Professor Welsing M. Pereira
www.professorwelsing.webnode.com

- Introdução
  - Processos de aplicação que rodam em máquinas diferentes se comunicam uns com os outros enviando mensagens para sockets.
  - Socket é a porta entre o processo da aplicação e o TCP.
  - O desenvolvedor da aplicação controla tudo que está no lado da camada de aplicação da porta; contudo, tem pouco controle do lado da camada de transporte (No máximo, poderá fixar alguns parâmetros do TCP, tais como tamanho máximo do buffer e tamanho máximo de segmentos).

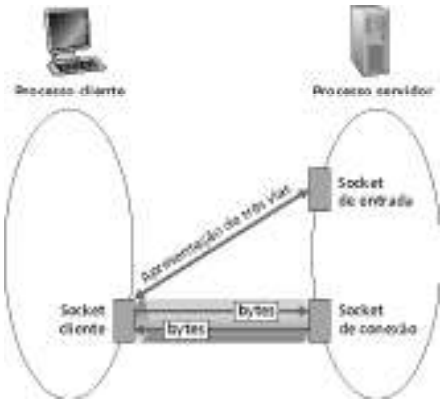
- Sockets



- Interação dos programas clientes e servidores
  - O cliente tem a tarefa de iniciar contato com o servidor.
  - O servidor deve estar pronto como um processo antes de o cliente tentar iniciar contato.
  - O servidor deve disponibilizar alguma porta (um socket) que acolha algum contato inicial do processo cliente.
  - Com o processo servidor em execução, o processo cliente pode iniciar uma conexão TCP com o servidor, o que é feito no programa cliente pela criação de um socket.
  - Quando cria seu socket, o cliente especifica o endereço do processo servidor, a saber, o endereço IP do hospedeiro servidor e o número de porta do processo servidor.
  - Com a criação do socket no programa cliente, o TCP no cliente inicia uma apresentação de três vias estabelece uma conexão TCP com o servidor.

Sockets, RPC, RMI-Java e CORBA

- Sockets
  - Durante a apresentação o processo servidor cria uma nova porta (um novo socket) dedicada àquele cliente específico.



Sockets, RPC, RMI-Java e CORBA

- RPC (Chamada de Procedimento Remoto)
  - Em 1970 foi proposto o conceito de ***chamada de procedimento remoto*** para proporcionar uma abordagem estruturada, de alto nível, à comunicação interprocessos em sistemas distribuídos.
  - Uma chamada a procedimento remoto permite que um processo que esteja em execução em um computador invoque um procedimento (ou função) de um processo que esteja em execução em outro computador.
  - Esse mecanismo pressupõe um modelo cliente/servidor:
    - O computador cliente que emite a chamada ao procedimento remoto envia seus parâmetros pela rede ao servidor onde reside o procedimento chamado.
    - A execução ocorre no servidor, e o resultado (o valor de retorno da função) então é transmitido pela rede ao cliente.

Sockets, RPC, RMI-Java e CORBA

- RPC (Chamada de Procedimento Remoto)
  - Para fornecer transparência aos programadores de sistemas distribuídos, a RPC introduz o conceito de stub.
  - Um ***stub*** prepara dados de saída para transmissão e traduz dados de entrada para que possam ser interpretados corretamente.
  - Para emitir uma RPC, o processo cliente faz uma chamada (passando os parâmetros apropriados) ao procedimento no ***stub do cliente***.
  - O stub do cliente executa a ***montagem de dados***, que empacota argumentos de procedimento juntamente com o nome do procedimento em uma mensagem para transmissão por uma rede.
  - O stub do cliente passa a mensagem (com parâmetros preparados) ao servidor.

Sockets, RPC, RMI-Java e CORBA

- RPC (Chamada de Procedimento Remoto)
  - Ao receber a mensagem do stub do cliente, o sistema operacional do servidor transmite a mensagem ao stub do servidor.
  - A mensagem é desmontada, e o stub envia os parâmetros ao procedimento local apropriado.
  - Quando o procedimento for concluído, o stub do servidor monta o resultado e o envia de volta ao cliente.
  - Por fim, o stub do cliente desmonta o resultado, notifica o processo e passa o resultado para ele.
  - Do ponto de vista do cliente, tudo isso parece apenas fazer uma chamada ao procedimento local e receber o resultado do retorno – os mecanismos da RPC fim ocultos.

Sockets, RPC, RMI-Java e CORBA

- RMI (Invocação a método remoto)
  - O protocolo RPC Java, conhecido como RMI, habilita um processo Java que está executando em um computador a invocar um método de um objeto em um computador remoto usando a mesma sintaxe de uma chamada a método local.
  - Similar a RPC, os detalhes da montagem de parâmetros e transporte de mensagens por RMI são transparentes para o programa que emite a chamada.
  - Um benefício que programadores Java implementam sistemas distribuídos sem ter de programar soques explicitamente.
  - Três camada distintas de software caracterizam a arquitetura RMI: a ***camada de sub/esqueleto***, a ***camada remota de referência (RRL)*** e a ***camada de transporte***.
  - a camada de stub/esqueleto contém estruturas de montagem e de parâmetros análogas às dos stubs de cliente e servidor da RPC.

Sockets, RPC, RMI-Java e CORBA

- RMI (Invocação a método remoto)
  - Um stub RMI é um objeto Java que reside na máquina cliente e que fornece uma interface entre o processo cliente e o objeto remoto.
  - Quando um processo cliente invoca um método em um objeto remoto, o método do stub é chamado primeiro.
  - O stub emprega ***serialização de objeto*** para criar sua mensagem montada, característica que permite que objetos sejam codificados em correntes de bytes e transmitidos de um espaço de endereçamento para outro.
  - Serialização de objetos habilita programas a passar objetos Java como parâmetros e receber objetos como valores de retorno.

Sockets, RPC, RMI-Java e CORBA

- RMI (Invocação a método remoto)
  - Uma vez serializados pelo stub, parâmetros são enviados ao componente RRL do sistema RMI do lado do cliente.
  - A RRL usa a camada de transporte para enviar a mensagem montada entre o cliente e o servidor.
  - Quando a componente da RRL do lado servidor recebe os parâmetros montados, ele os dirige ao esqueleto, que desmonta os parâmetros, identifica o objeto sobre o qual o método deve ser invocado e chama o método.
  - Ao concluir o método, o esqueleto monta o resultado e o devolve ao cliente via RRL e stub.

Sockets, RPC, RMI-Java e CORBA

- CORBA (Common Object Request Broker Architecture)
  - É um padrão aberto, de ampla aceitação, elaborado para habilitar interoperação entre programas em sistemas heterogêneos, bem como homogêneos.
  - Similar a RMI, CORBA suporta objetos como parâmetros ou valores de retorno em procedimentos remotos durante comunicação interprocessos.
  - Entretanto, diferentemente da RMI (que é baseada em Java), CORBA é independente de linguagem de sistema, o que significa que aplicações escritas em linguagens de programação diferentes e em sistemas operacionais diferentes operam entre si por meio de acesso a um núcleo comum da arquitetura CORBA.

Sockets, RPC, RMI-Java e CORBA

- CORBA (Common Object Request Broker Architecture)
  - A estrutura dos sistemas distribuídos baseados em CORBA é relativamente simples.
  - O processo no cliente passa a chamada a procedimento juntamente com os argumentos requeridos ao stub do cliente.
  - O stub do cliente monta os parâmetros e envia a chamada ao procedimento por meio de seu ***agente de solicitação de objetos*** (Object Request Broker – ORB), que se comunica com o ORB do servidor.
  - O ORB no servidor então passa a chamada a procedimento ao esqueleto do servidor, que desmonta os parâmetros e passa a chamada ao procedimemnto remoto.

Sockets, RPC, RMI-Java e CORBA

- CORBA (Common Object Request Broker Architecture)
  - CORBA fornece a seus usuários independência de linguagem por intermédio da IDL (Interface Definition Language – linguagem de definição de interface).
  - A IDL permite que programadores definam estritamente os procedimentos que podem ser chamados no objeto, o que é conhecido como a interface daquele objeto.
  - Aplicações escritas em qualquer linguagem podem comunicar-se por meio de CORBA seguindo a especificação IDL de cada objeto.

Sockets, RPC, RMI-Java e CORBA

- Comparações:
  - Sockets
    - RMI é muito mais simples do que fazer troca de mensagens usando sockets porque os detalhes do estabelecimento das conexões entre os hosts e a transferência de dados entre eles são escondidos nas classes RMI.

Sockets, RPC, RMI-Java e CORBA

- RPC
  - Faz quase o mesmo que a RMI.
  - As diferenças são:
    - O RPC só pode enviar tipos de dados primitivos enquanto RMI pode enviar objetos.
    - RPC é independente da linguagem e RMI é limitado a programas escritos em Java.
    - RPC não é bem adaptado para programação orientada a objetos.

Sockets, RPC, RMI-Java e CORBA

- CORBA
  - É a solução mais geral para objetos distribuídos.
  - CORBA permite que objetos escritos em linguagens diferentes comuniquem-se entre si.
  - Java “adapta-se” a CORBA através da Java IDL (Interface Definition Language).