Você sabia que seu material didático é interativo e multimídia?
Ele possibilita diversas formas de interação com o conteúdo, a qualquer hora e de qualquer lugar. Mas na versão impressa, alguns conteúdos interativos são perdidos, por isso, fique atento! Sempre que possível, opte pela versão digital.
Bons estudos!



# Sistemas Distribuídos

Utilizando Sockets com Java

Unidade 4 – Seção 2

Nessa webaula vamos estudar dois tipos de protocolos: o TCP e o UDP. Em seguida, vamos dar início ao conceito sobre a funcionalidade sockets

## Protocolo TCP e UDP

A comunicação entre máquinas em rede é essencial para diversas aplicações que utilizamos, tanto dentro de empresas, quanto no nosso dia a dia pessoal. Para facilitar o entendimento dos diferentes elementos envolvidos em uma comunicação em rede de computadores, frequentemente utilizamos o modelo de comunicação de 7 camadas ISO/OSI como referência (MAIA, 2013). Nesse modelo, a comunicação dos dados de uma máquina para outra ocorre a partir da quarta camada (a chamada camada de transporte). Nessa camada, destacam-se dois protocolos muito utilizados para realizar tal comunicação: o protocolo TCP e o protocolo UDP.

O Protocolo de Controle de Transmissão – TCP (do inglês, *Transmission Control Protocol*), é um protocolo utilizado como base para comunicação entre máquinas, onde, caso haja alguma perda de informação, durante a transmissão, aquela informação seja retransmitida (MAIA, 2013). Por essa razão, é dito que o TCP é um protocolo orientado à conexão. Assim, esse é o protocolo de escolha para comunicações que não necessitem de uma transmissão em tempo real, ou seja, que não são muito sensíveis a atrasos, já que essa retransmissão leva mais tempo para ser realizada.

Por outro lado, o protocolo UDP (do inglês, *User Datagram Protocol*), é um protocolo utilizado como base para comunicação entre máquinas nas quais é importante que o atraso entre o envio e o recebimento da mensagem seja minimizado. Por essa razão, é dito que o UDP é um protocolo que não é orientado à conexão (referido como *connectionless*). Assim, esse é o protocolo de escolha para comunicações que necessitem de uma transmissão em tempo real, ou seja, não podem ocorrer atrasos, já que esse protocolo não retransmite a informação, como é o caso para a maioria das aplicações de tempo real (*real time*).

### Comunicação através de *sockets*

Os sockets utilizam o TCP (ou UDP) para realizar a comunicação entre aplicações que estejam sendo executadas em um sistema operacional, razão pela qual essa comunicação é chamada de interprocessos. Além disso, os *sockets* abstraem, do programador, a necessidade de um aprofundamento nas camadas mais inferiores de comunicação (camadas 1 a 3 do modelo de referência ISO/OSI).

Para que essa abstração possa ocorrer, existem funcionalidades (por vezes chamadas de primitivas) que normalmente são fornecidas por qualquer implementação de *socket*; em qualquer linguagem de programação orientada a objetos, essas funcionalidades representam métodos, já implementados, em determinadas classes relativas à comunicação via rede. Um resumo dessas funcionalidades é apresentado na tabela.

| Funcionalidade | Significado |
|---|---|
| *Socket* | Cria um novo terminal de comunicação. |
| *Bind* | Atrela um endereço IP local a um *socket*. |
| *Listen* | Aviso de que o *socket* está aceitando conexões. |
| *Accept* | Aguarda o recebimento de uma solicitação de conexão. |
| *Connect* | Ativamente tenta estabelecer conexão com um *socket*. |
| *Send* | Envia dados através de uma conexão previamente estabelecida. |
| *Receive* | Recebe dados através de uma conexão previamente estabelecida. |
| *Close* | Libera a conexão. |

Em termos práticos, um *socket* é uma combinação de endereço IP e porta de comunicação. Conforme observa Coulouris et al. (2013), para um processo de enviar e receber mensagens, seu *socket* precisa estar atrelado a uma porta e a um endereço IP roteável na máquina onde esse processo está sendo executado.

Observe que existe uma ordem “natural” de chamada dessas primitivas, tanto por uma máquina atuando como cliente, quanto por uma máquina atuando como servidor, para que a comunicação entre essas máquinas possa ocorrer.

Nessa webaula tivemos a introdução do conceito sobre a funcionalidade socket, acesse o livro didático para continuar estudando esse conceito através de um exemplo na linguagem Java.