Utilizando sockets com Java

Sistemas Distribuídos

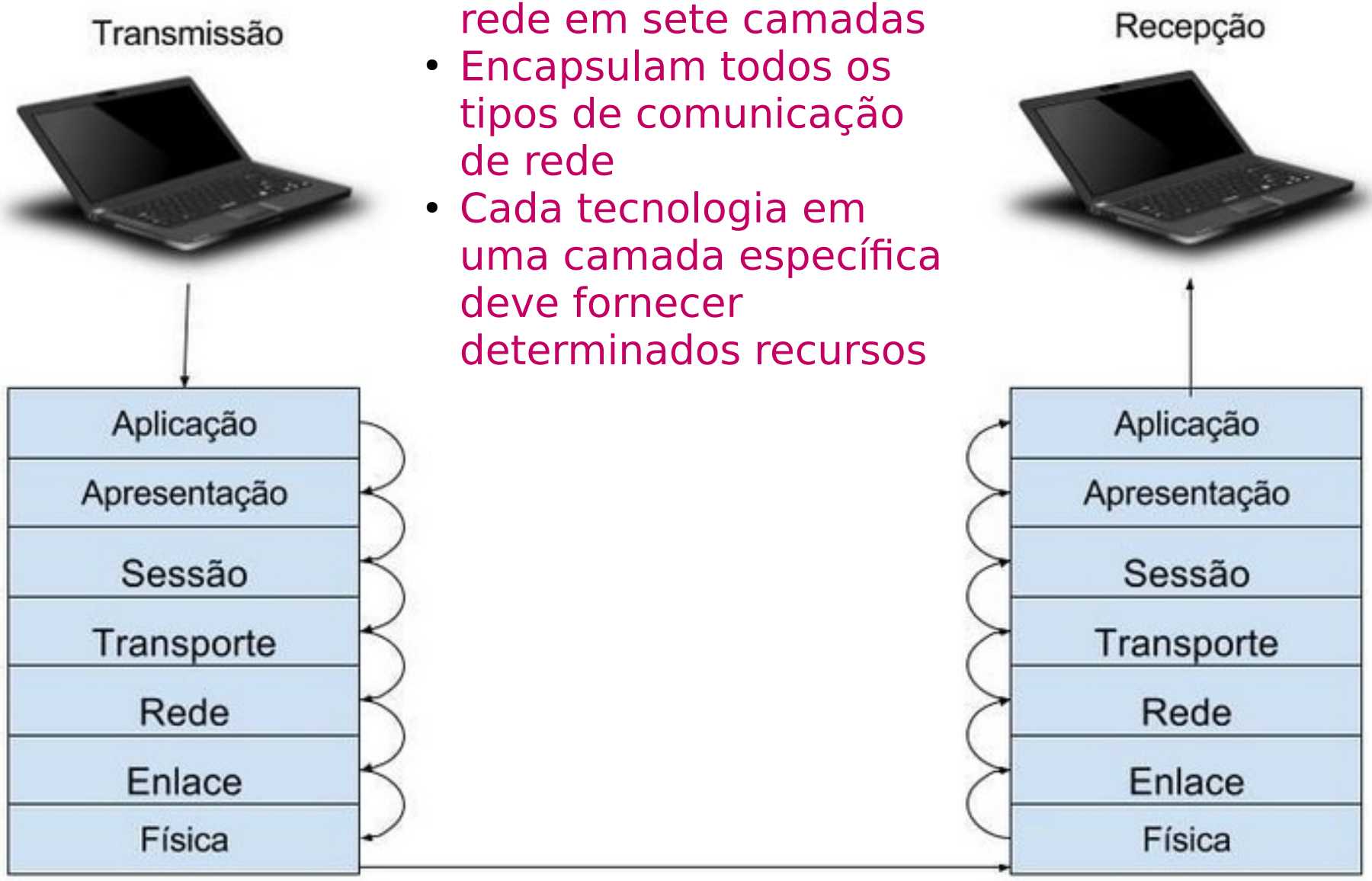
Eduardo Furlan Miranda

2025-10-07

Adaptado de: PEREIRA, C. S. Sistemas Distribuídos. Londrina: EDE SA, 2019. ISBN 978-85-522-1443-4.

7 camadas ISO/OSI

- Estrutura conceitual que divide as funções de comunicação de rede em sete camadas
- Encapsulam todos os tipos de comunicação de rede
- Cada tecnologia em uma camada específica deve fornecer determinados recursos



- A comunicação dos dados de uma máquina para outra ocorre a partir da quarta camada (a chamada camada de transporte)
- Destacam-se dois protocolos muito utilizados para realizar tal comunicação: o protocolo TCP e o protocolo UDP
- TCP: caso haja alguma perda de informação durante a transmissão, a informação é retransmitida
- UDP: mais simples, rápido, porém não retransmite

Sockets

- Mecanismo de comunicação de rede
- Enviar e receber dados de forma bidirecional entre diferentes processos, na mesma máquina ou em diferentes
- Usa o protocolo TCP para realizar a comunicação entre aplicações que estejam sendo executadas em um sistema operacional
- Usa IP:Porta (ex.: 192.168.1.10:8080)
- Modelo cliente-servidor

Primitivas de sockets TCP

Significado	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.

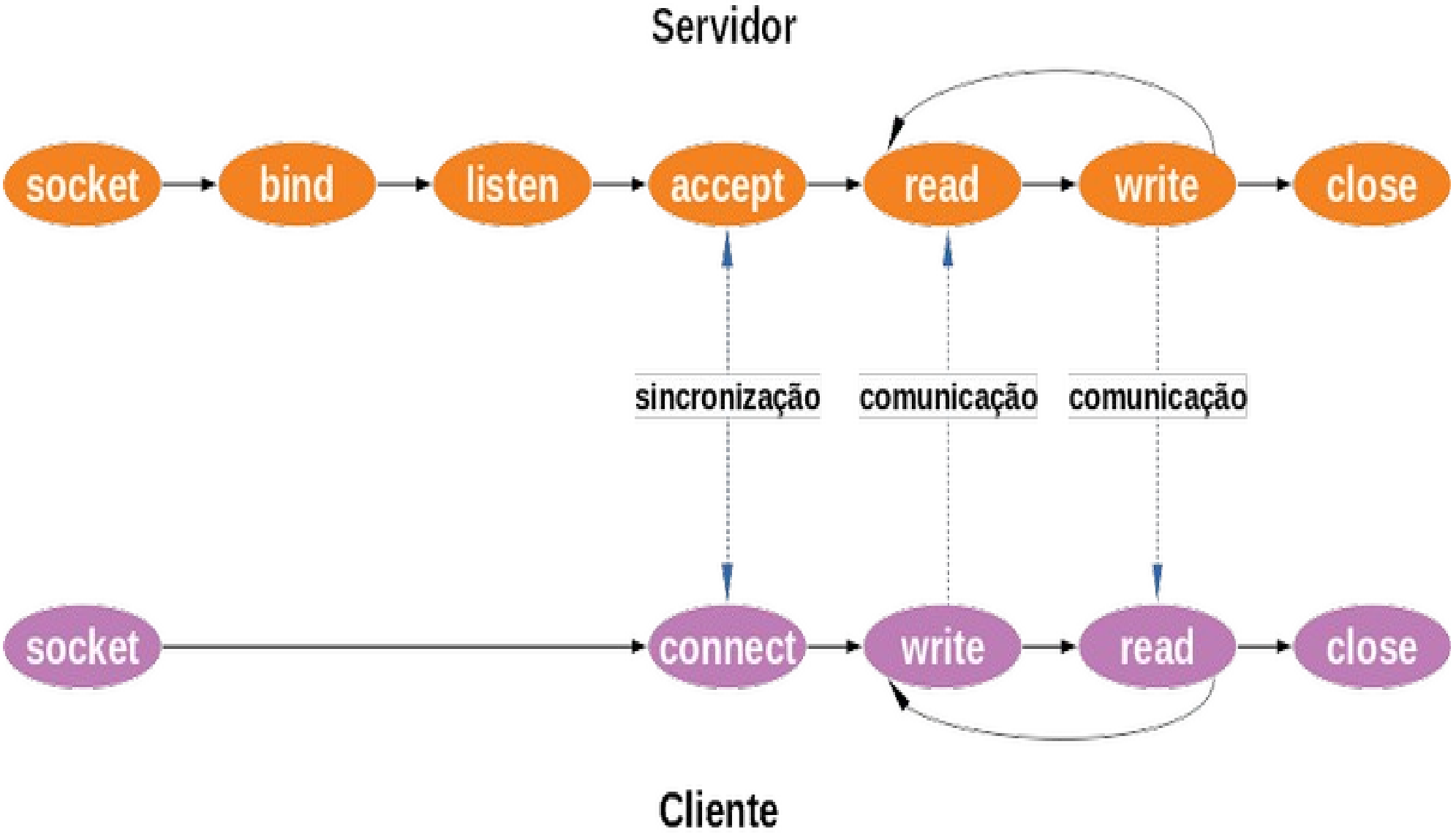
No servidor

- A primitiva “socket” cria um novo terminal de comunicação
- Internamente o sistema operacional aloca os recursos necessários
- “bind” associa um socket específico ao servidor, de forma que o sistema operacional saiba
- “listen” faz com que o sistema operacional reserve recursos
- “accept” fica aguardando o estabelecimento de uma conexão
  - quando esta ocorre, o sistema operacional retorna um socket com as mesmas características daquele criado pelo servidor

Nos clientes

- Cria-se um socket, mas não há necessidade de especificar uma porta de comunicação
- “connect” recebe a informação de qual protocolo de transporte foi adotado pelo servidor e fica aguardando o estabelecimento de uma conexão
- “send” e “receive” troca informações ou mensagens
- “close” explicitamente encerra e libera os recursos

Processo de comunicação via sockets TCP



Servidor

```java
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

public class Servidor {

	public static void main(String[] args) {
		Socket soc = null;
		ServerSocket socServidor = null;
		try {
			socServidor = new ServerSocket(5001);
			soc = socServidor.accept();
			DataInputStream recebido = new DataInputStream(soc.getInputStream());
			DataOutputStream enviado = new DataOutputStream(soc.getOutputStream());
			System.out.println("(cliente): " + recebido.readUTF());
			enviado.writeUTF("O servidor recebeu sua mensagem.");
		} catch (IOException ex) {
			System.err.println("Falha na conexão");
		} finally {
			try {
				soc.close();
				socServidor.close();
			} catch (IOException e) {
				System.err.println("Falha ao encerrar a conexão");
			}
		}
	}
}
```

Cliente

```java
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;

public class Cliente {

	public static void main(String[] args) {
		Socket soc = null;
		try {
			soc = new Socket("127.0.0.1", 5001);
			DataInputStream recebido = new DataInputStream(soc.getInputStream());
			DataOutputStream enviado = new DataOutputStream(soc.getOutputStream());
			enviado.writeUTF("Aqui é um cliente falando...");
			System.out.println("(servidor): " + recebido.readUTF());
		} catch (IOException ex) {
			System.err.println("Falha na inicializar o servidor");
		} finally {
			try {
				soc.close();
			} catch (IOException e) {
				System.err.println("Falha ao encerrar a conexão");
			}
		}
	}
}
```