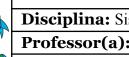
# UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO Campus Pau dos Ferros



**Disciplina:** Sistemas operacionais

**Professor(a):** Reudisman Rolim de Souza

**Discentes:** Kauê Valentim de Carvalho e Jean Richardson Gonçalves Marcos

Curso: BACHARELADO EM TECNOLOGIA Semestre:2025.1

DA INFORMAÇÃO

Comunicação entre Processos

### 1. Introdução

Este projeto representa uma simulação prática de comunicação entre processos, utilizando sockets, na linguagem de programação orientada á objetos "java". A aplicação foi construída visando o aprofundamento no entendimento dos conceitos de comunicação entre cliente e servidor.

#### 2. Objetivo

O principal objetivo do projeto é permitir que um cliente envie uma mensagem para um servidor, e que esse servidor, ao recebê-la, responda de forma apropriada.

#### 3. Descrição da Implementação

A aplicação é composta por duas classes principais: Server.java e Cliente.java.

#### Servidor (Server.java)

- Inicia um ServerSocket na porta 8000.
- Aguarda conexões de clientes com accept().
- Cria fluxos de entrada e saída (BufferedReader e PrintWriter).
- Lê uma mensagem enviada pelo cliente.

- Responde com a mensagem "Hello world".
- Fecha a conexão e libera os recursos.

#### Cliente (Cliente.java)

- Conecta-se ao localhost pela porta 8000 usando Socket.
- Cria fluxos de entrada e saída para comunicação.
- Envia a mensagem "Obrigado por me deixar entrar".
- Lê e imprime a resposta do servidor.
- Fecha a conexão corretamente.

#### 4. Detalhes Técnicos

O projeto utiliza as bibliotecas **java.net** e **java.io**, fundamentais para conexões de rede e manipulação de dados.

#### **Principais Classes Utilizadas:**

- java.net.ServerSocket: cria o ponto de escuta no servidor.
- java.net.Socket: cria o canal de comunicação entre cliente e servidor.
- PrintWriter: envia texto ao socket de forma simples e eficiente.
- InputStreamReader + BufferedReader: lê texto do socket de forma estruturada.

#### Fluxo da Comunicação:

- 1. O servidor é iniciado e escuta na porta 8000.
- 2. O cliente é executado e se conecta ao servidor.

- 3. O cliente envia uma mensagem.
- 4. O servidor recebe e imprime essa mensagem.
- 5. O servidor responde.
- 6. O cliente recebe a resposta.
- 7. Ambos encerram a conexão.

## 5. Conclusão

Conclui-se que é possível realizar a comunicação entre dois processos distintos por meio da utilização de sockets.