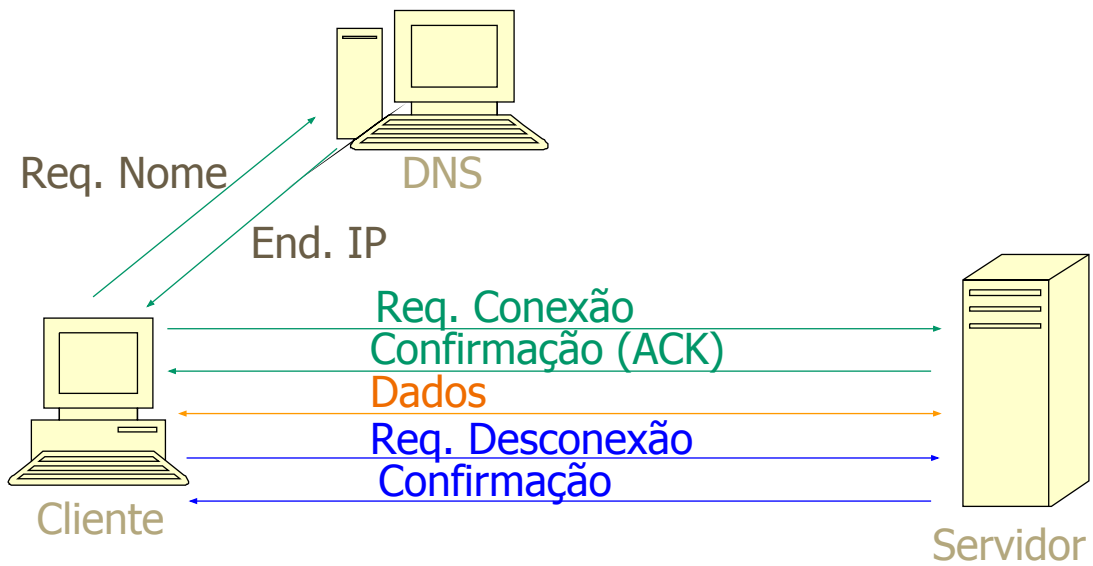
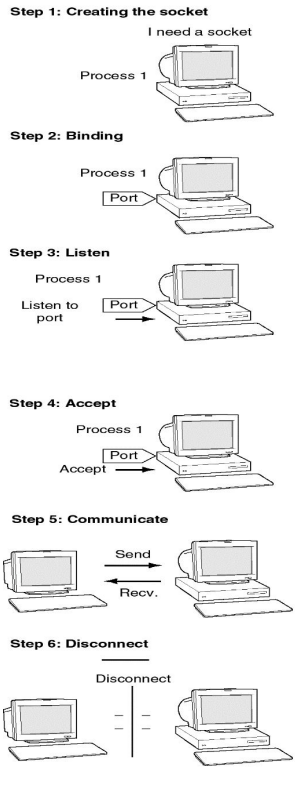
Protocolos TCP

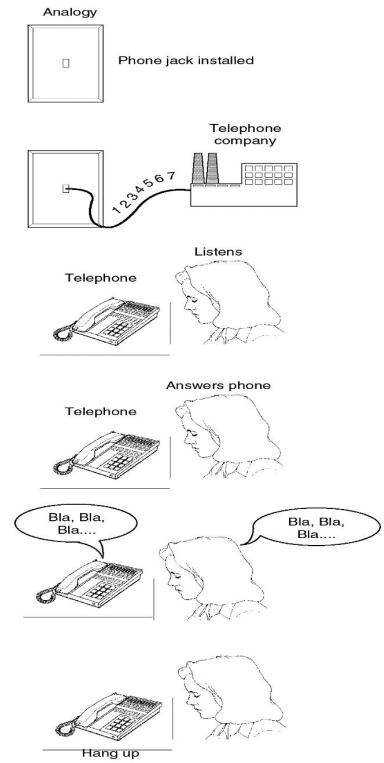
- ***Transmission Control Protocol (TCP):***
- Protocolo orientado à conexão:
  - Exige o estabelecimento de um canal lógico para iniciar a transmissão de dados, em 3 fases:
    - Fase de conexão
    - Fase de dados
    - Fase de desconexão
- Exemplos de aplicação:
  - TELNET, Web Browser, ...

Etapas em uma Conexão TCP

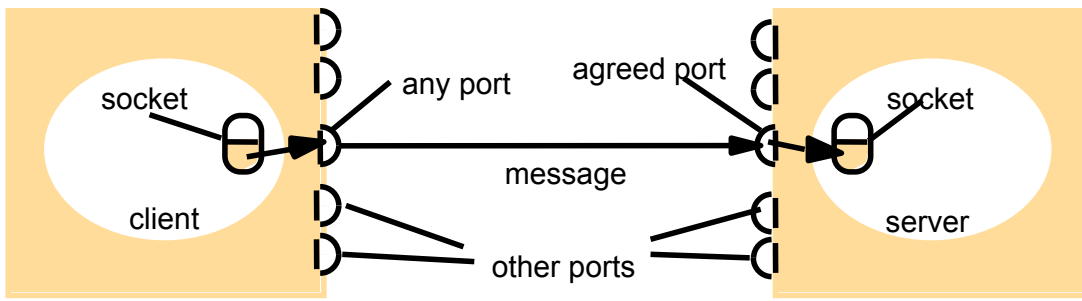


Socket: Uma analogia





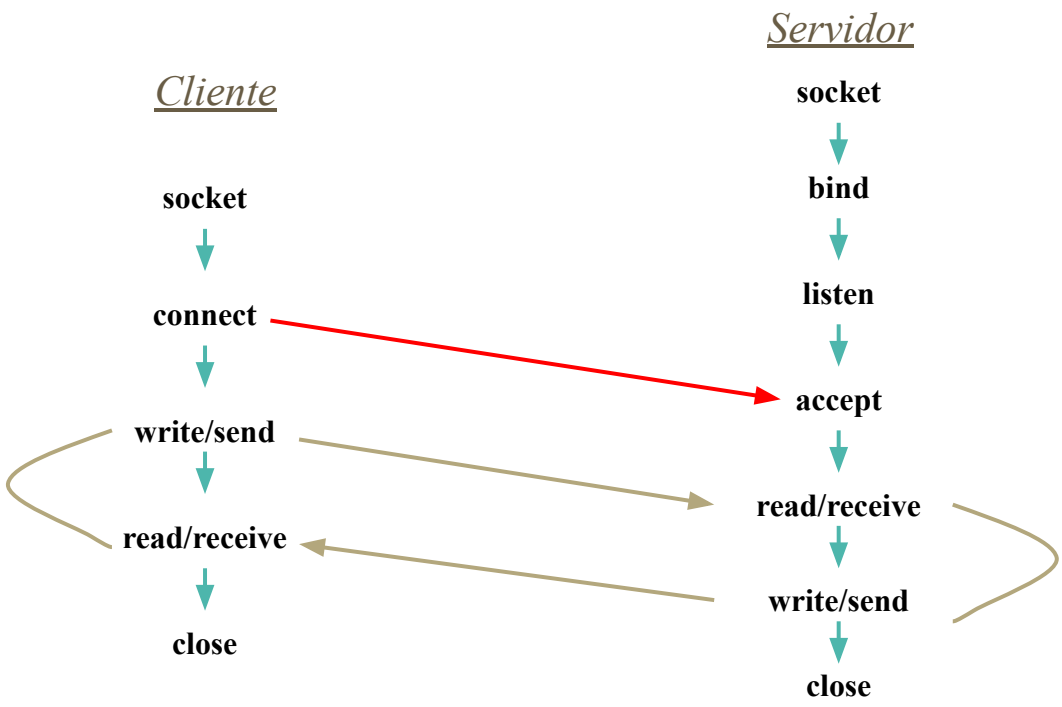
Sockets e Portas



Internet address = 138.37.94.248

Internet address = 138.37.88.249

Socket: Comunicação C/S

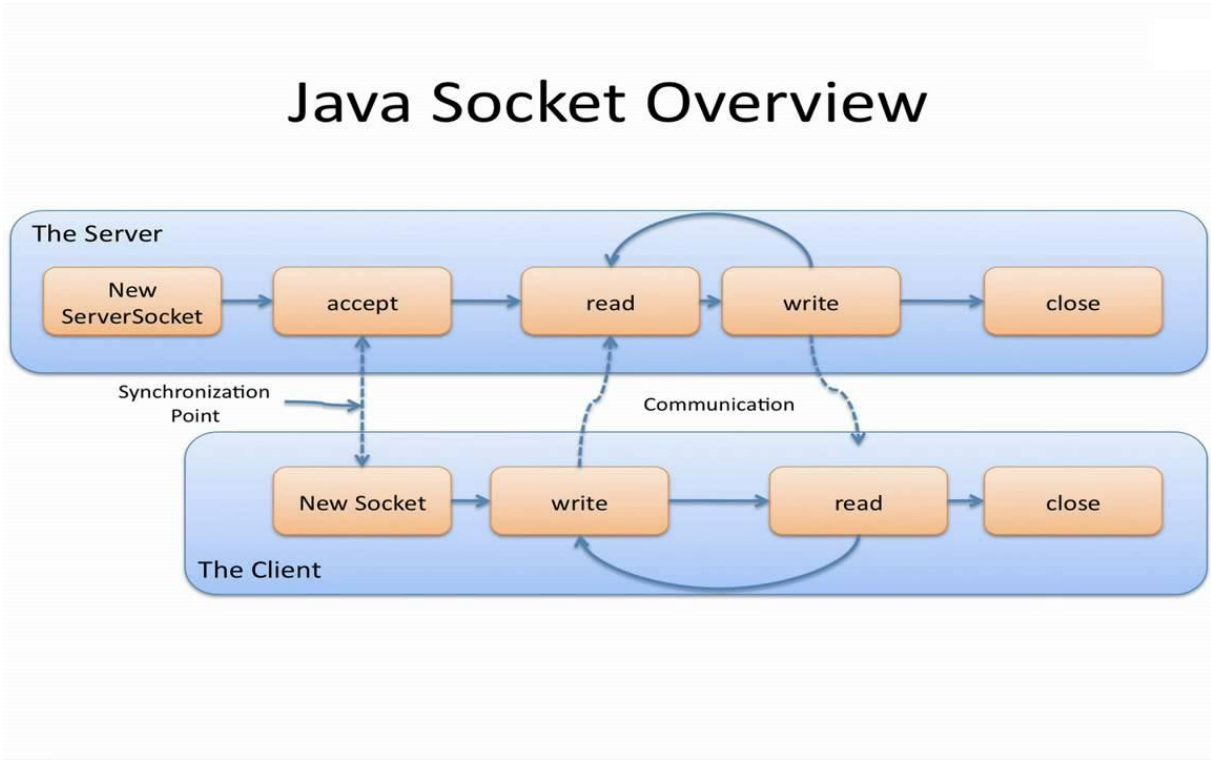


Socket: Comunicação C/S

- **Servidor:**
  - Efetua a criação de um *Socket*;
  - Associa o *Socket* a um endereço local;
  - Aguarda por conexões da parte cliente;
  - Aceita conexões;
  - Lê requisições;
  - Opcionalmente envia resposta;
  - Fecha o *Socket*.

Sockets em JAVA

Modelo



Tipos de Sockets em Java

- Pacote *java.net*
  - *java.net.Socket*
  - *java.net.ServerSocket*
- *Socket* - usado em cada lado do canal de comunicação bidirecional.
- *ServerSocket* - responsável por ficar aguardando pedidos de conexão dos clientes

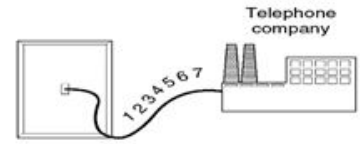
Tipos de Sockets em Java

- *Socket* e *ServerSocket* são sockets do tipo *StreamSocket*
  - Utilizam protocolo TCP
  - Orientado à conexão (o servidor precisa aceitar o pedido de conexão do cliente)
- Outro tipo é o *DatagramSocket*
  - Utiliza protocolo UDP
  - Não é orientado à conexão

Sockets em Java (Server)

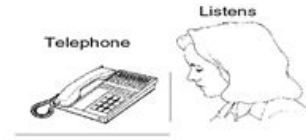
- *ServerSocket srv = new ServerSocket(9876);*

1

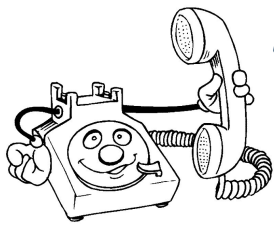


- Socket cliente = srv.accept();

2



3

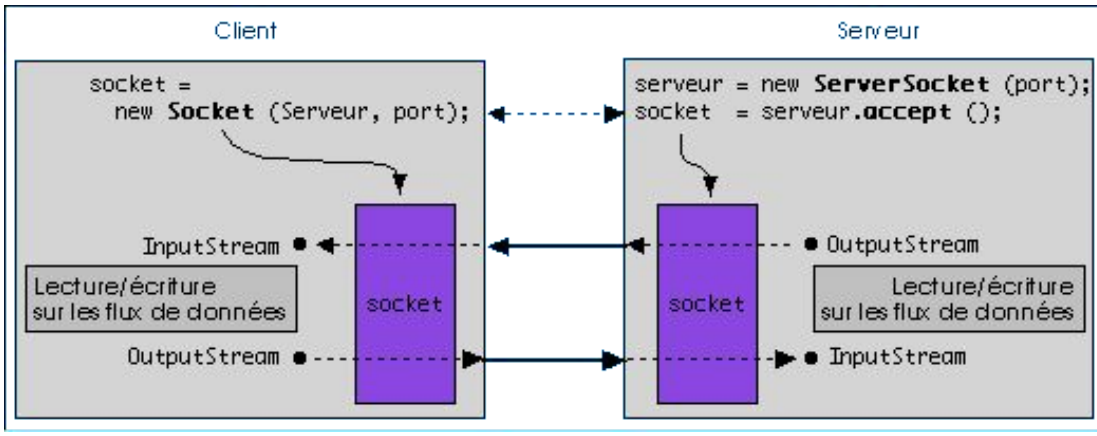


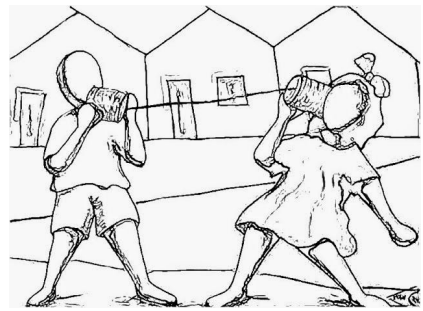
Sockets em Java (Cliente)

- *Socket sc = new Socket("127.0.0.1" , 9876);*

1

Leitura/Escrita em Sockets





Leitura/Escrita em Sockets

- Sockets enviam e recebem dados na forma de *bytes*
- Cada conjunto de bytes é chamado *stream*
- Utilizam os seguintes métodos
  - getInputStream
    - read
  - getOutputStream
    - write