Trabalho

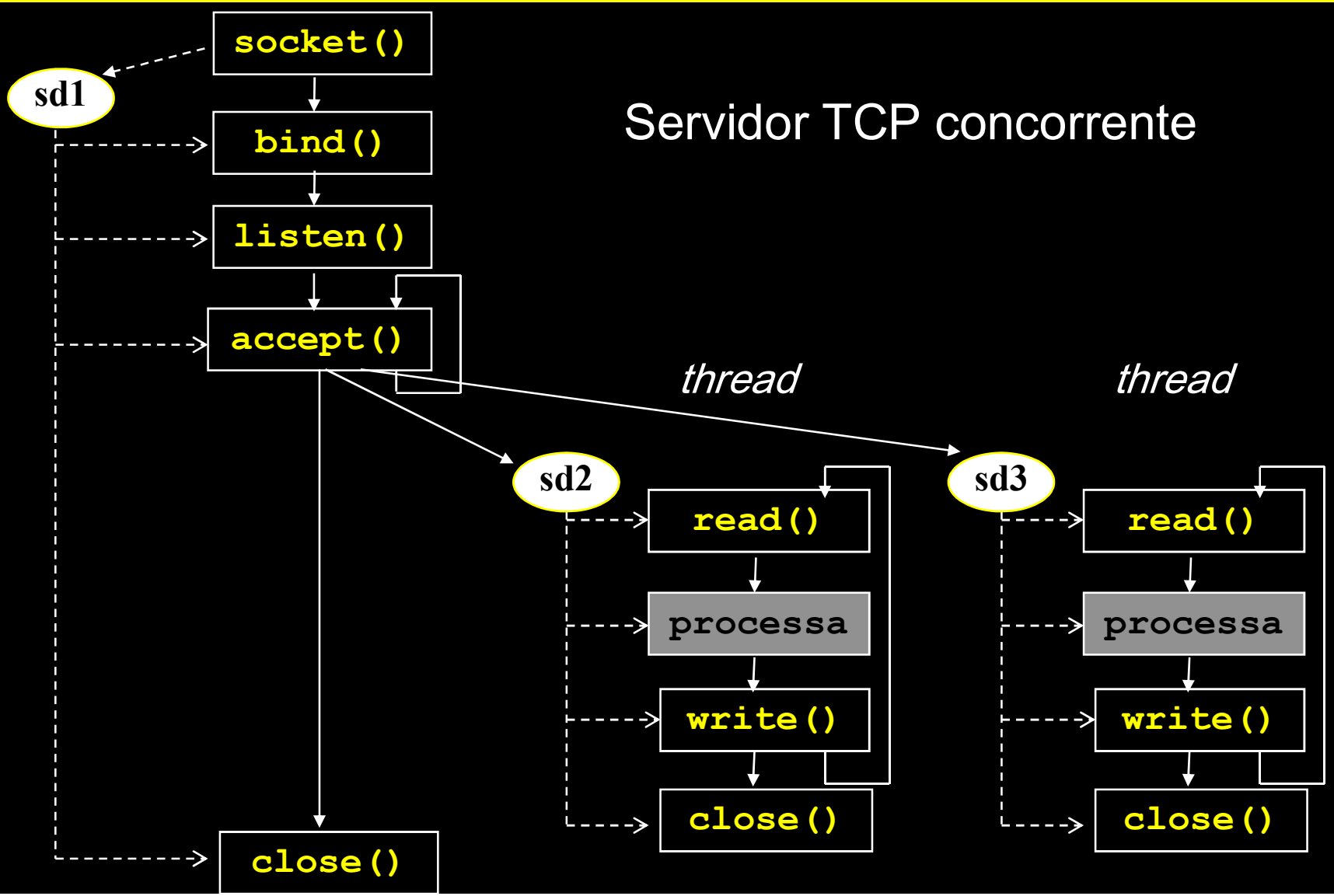
Servidor TCP Concorrente

Programação Sockets

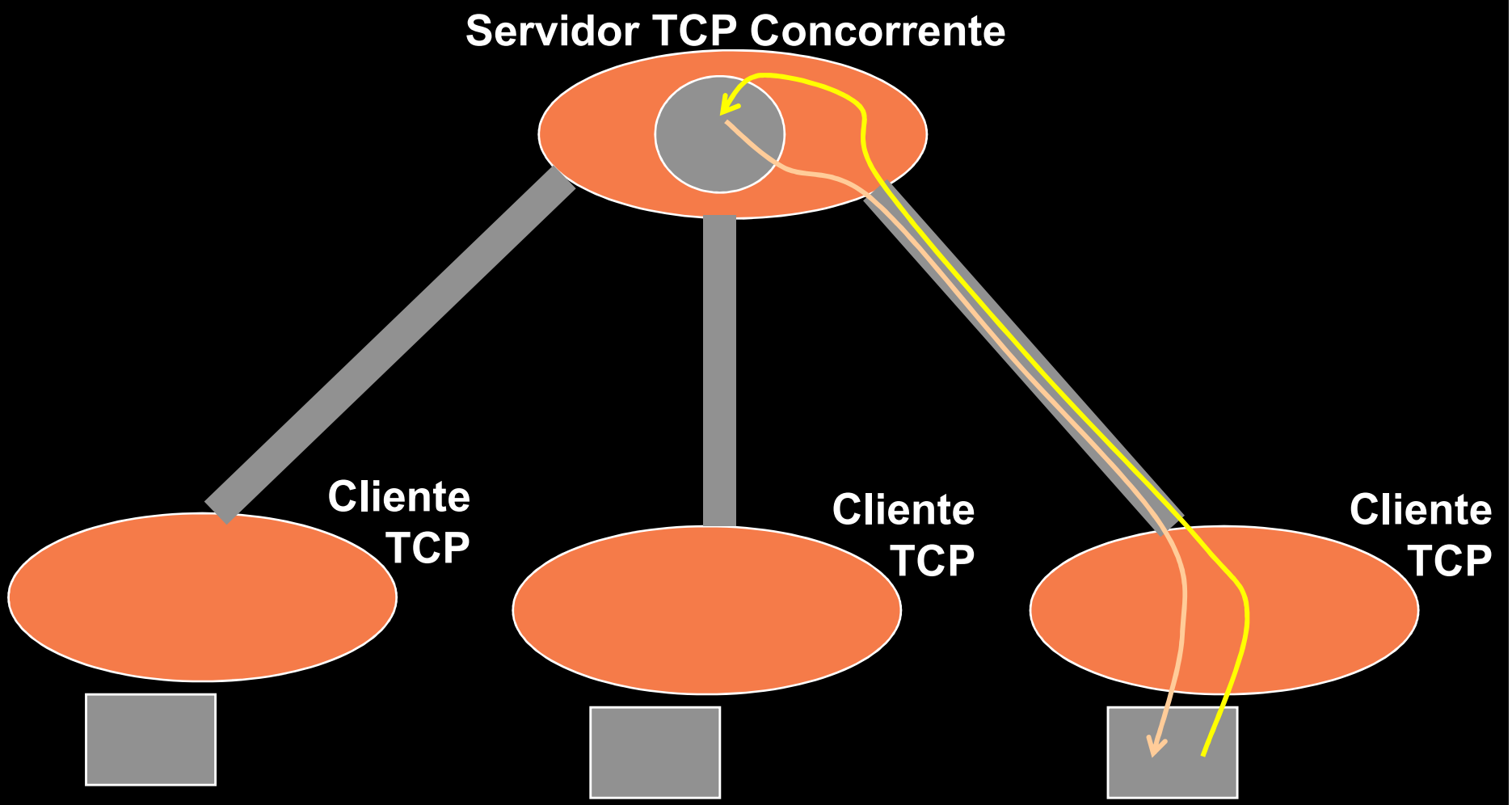
Trabalho: Servidor TCP concorrente

- **Objetivo:**
  - Desenvolvimento de um programa servidor TCP concorrente
  - Deve disponibilizar informações sobre o servidor em função do pedido: “datahora”: data e hora, “temperatura”: temperatura do processador, “memória”: dados de memória e “disco”: dados de disco.
- **Grupo**
  - Cada grupo de 2 pessoas
- **Formato do trabalho**
  - Formato eletrônico, depositado no moodle
  - Página de rosto informando:
    - Nome da disciplina, título do trabalho e nome dos autores
- **Entrega:**
  - Data entrega: **17 de junho**
  - Execução do programa durante a aula
  - Serão descontados 2 pontos da nota para cada dia de aula em atraso

Trabalho: Servidor TCP concorrente



Chat UDP


Servidor TCP Concorrente
Cliente TCP
Cliente TCP
Cliente TCP

Trabalho: Servidor TCP concorrente

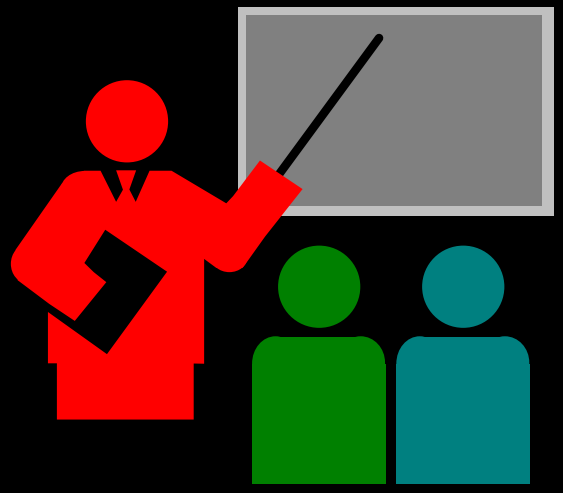
- **Servidor pode ter diversas conexões estabelecidas com um número máximo de C clientes (C é parâmetro do programa).**
- **Fluxo:**
  - Processo cliente solicita conexão ao servidor
  - Após estabelecimento de conexão cliente pode encaminhar quantos comandos quiser a qualquer momento.
    - Cliente: envia comando
    - Servidor: envia resposta (conteúdo ASCII)
  - Comando “exit” permite ao cliente solicitar o encerramento da conexão
    - Servidor, ao receber o comando “exit”
      - Encaminha a mensagem ao cliente “Servidor encerrando a conexão”.
      - Encerra a conexão TCP.
- **Protocolo de comunicação**
  - Definido pelo grupo

Trabalho: Servidor TCP concorrente

- **Comandos para o servidor:**

date	Data e hora corrente	Comando date
version	Versão do kernel	Arquivo /proc/version
cpu	Dados da CPU	Arquivo /proc/cpuinfo
memory	Dados da memória	Arquivo /proc/meminfo
partitions	Partições de disco	Arquivo /proc/partitions
Interfaces	Interfaces de rede	Comando /sbin/ifconfig
route	Tabela de roteamento	Arquivo /proc/route
exit	Encerra a conexão	Responde confirmação

Dicas



Dicas

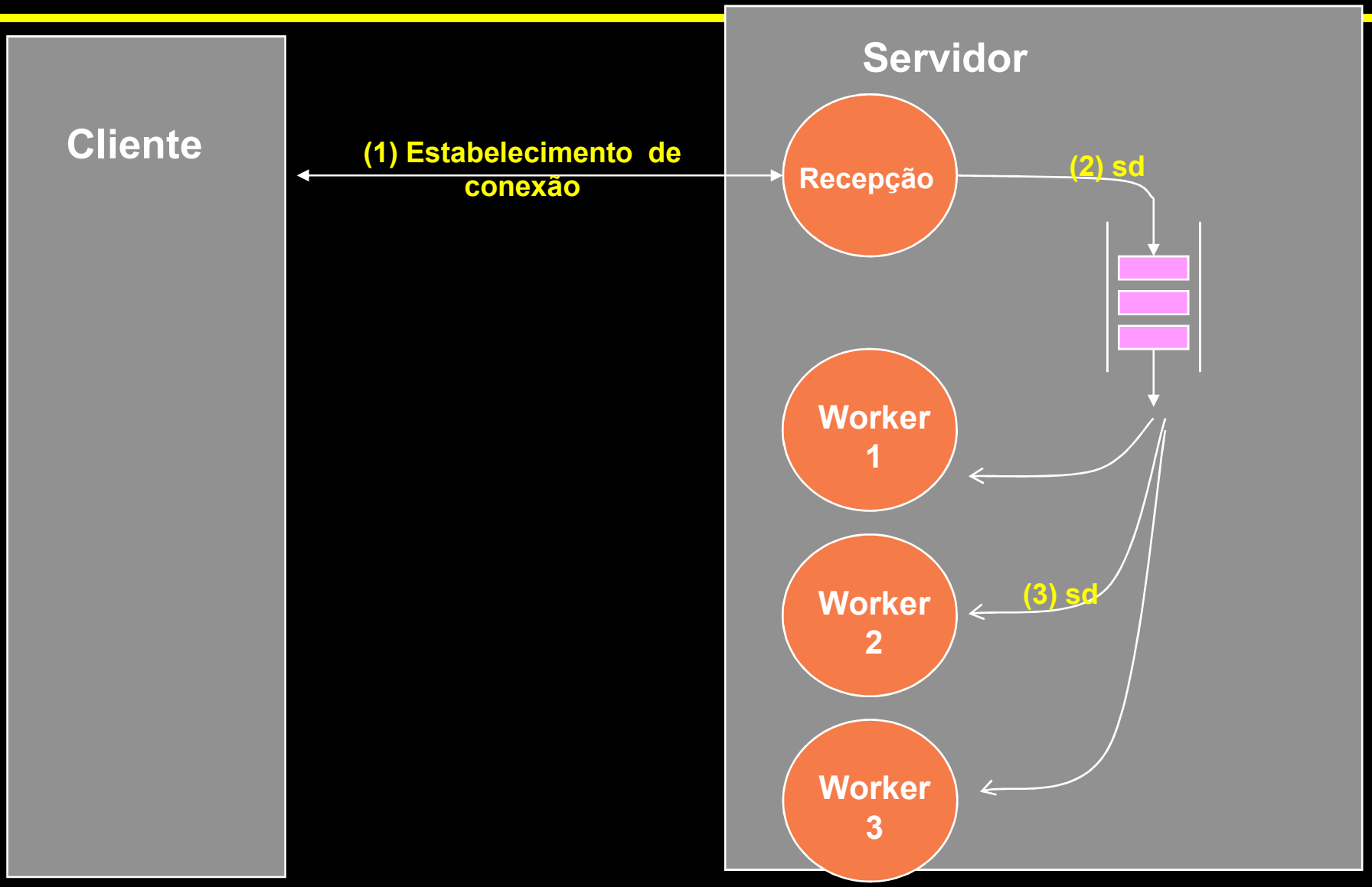
- **Para testar**
  - Utilizar o programa cliente echo TCP
  - Importante alterar o buffer de recepção, pois os dados recebidos podem ser grandes.

Dicas

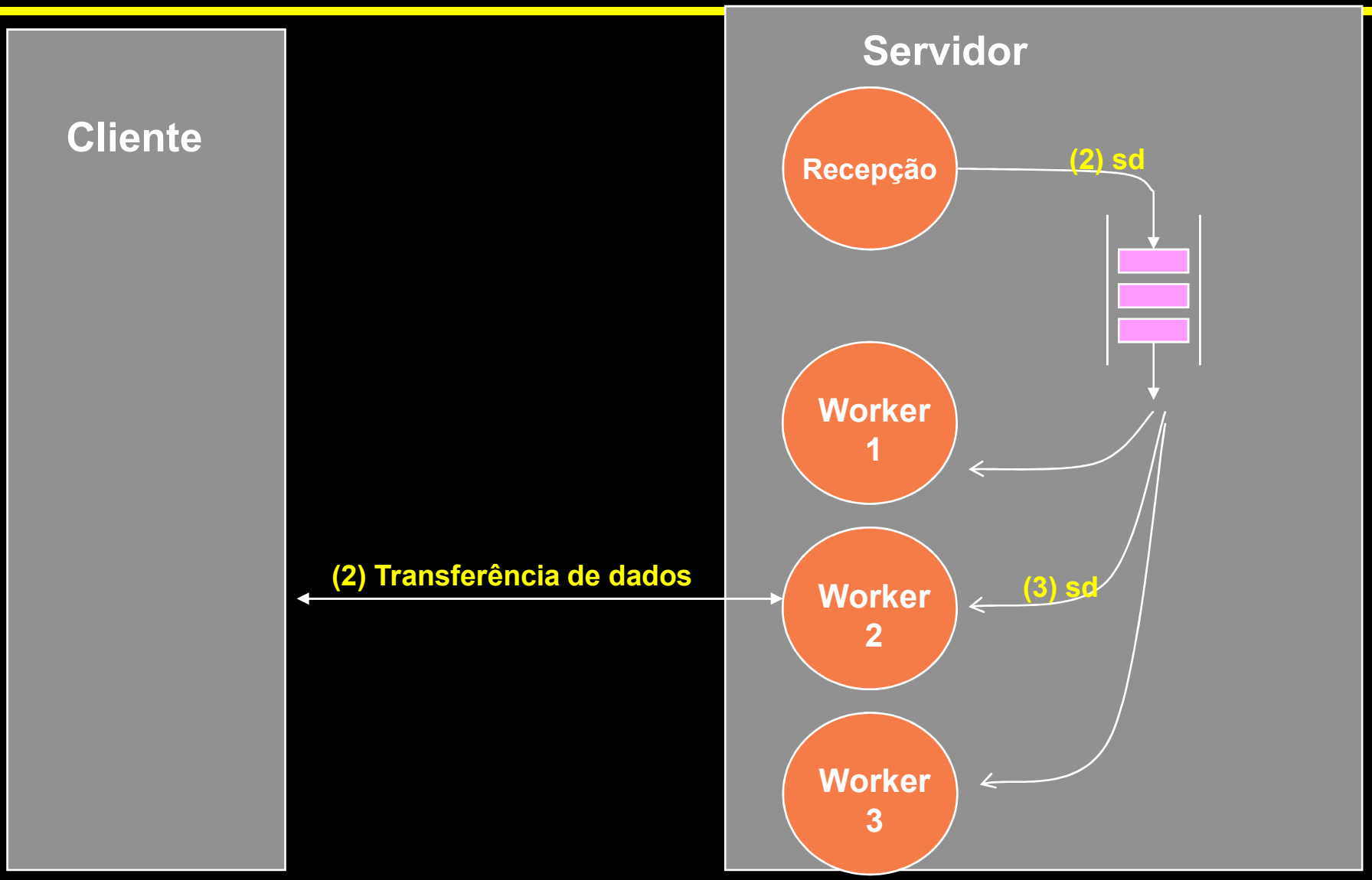
- **Modelo produtor-consumidor.**
  - Produtor:
    - Thread de recepção:
    - Produz novas conexões (produz socket descriptors qu são números inteiros).
  - Consumidores:
    - Threads de trabalho (worker)
    - Consomem conexões (socket descriptors).
    - O consumidor, ao receber o socket descriptor fica responsável pela interação com o cliente até que a conexão até o encerramento da conexão.

→ Usar a solução do problema Produtor-Consumidor com fila sincronizada por semáforos !!!

Exercício


Cliente
(1) Estabelecimento de conexão
Servidor
Recepção
(2) sd
Worker 1
Worker 2
(3) sd
Worker 3

Exercício


Cliente
Servidor
Recepção
(2) sd
Worker 1
Worker 2
Worker 3
(2) Transferência de dados
(3) sd

Dicas

- **Função transferfile()**
  - Para transferência do conteúdo de arquivo para uma conexão TCP utilize a função transferfile():

    int transferfile(char *path,int output_fd)

  - A função transferfile realiza a leitura do conteudo de um arquivo, identificado por seu caminho (path), transferindo seu conteudo para outro arquivo ou socket identificado pelo descritor de arquivos "output_fd“.