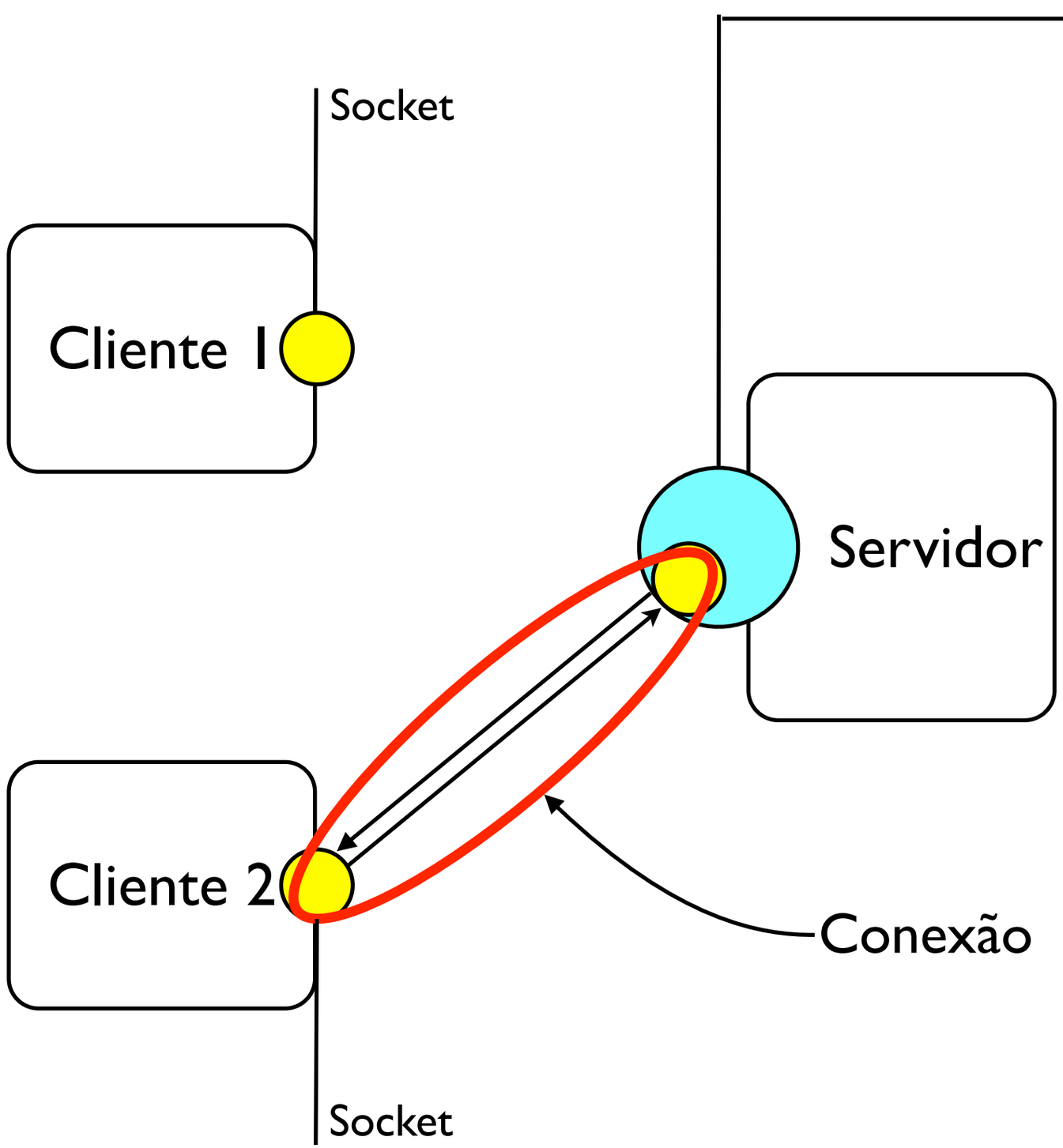
Sistemas Distribuídos

Universidade do Minho

2011/2012

Aulas anteiores: Sincronização

- Exclusão mútua: garantir que dois processos ou threads não tenham acesso simultaneamente a um recurso partilhado.
- Semáforos, barreiras, etc.
- Proteger secções críticas do código.
- Variáveis de condição.
  - suspensão/retoma de execução dentro de zona crítica;
- Starvation
- ...



Server Socket TCP/IP:

- endereço (ip);
- porto 16 bits-> distinguem serviços na mesma máquina;
  - 0–1023 são standard ex: http 80
  - 1024–49151
  - 49152-65535 dinâmicos

- Servidor fica à espera de ligações;
- quando o cliente se liga é estabelecida uma conexão, bidireccional;
- Socket representa um extremo de uma conexão.

- Classes e métodos relevantes:
  - java.net.io.*, java.net.ServerSocket
  - – métodos relevantes: ServerSocket(), accept(), close()
  - – outros métodos: setReuseAddress(), bind()
  - java.net.Socket
    – métodos relevantes: Socket(), connect(), read(),write(),getInputStream(),getOutputStream()
  - – outros métodos: shutdowInput(), shutdownOutput()

Aula 7: Socket JAVA

Cliente

Esqueleto: JAVA:

socket()
connect() — `Socket socket = new Socket(remotehost,port);`

while ()

write() — `out.write(...);`
read() — `out.flush();`

close()

```
socket.shutdownInput();
socket.shutdownOutput();
socket.close();
```

Servidor

Esqueleto: JAVA:

socket()
bind() → `ServerSocket sSock = new ServerSocket(porto);`

listen()
while ()
accept()

```
while (true){ //para aceitar conexões indefinidamente
Socket sock = sSock.accept()
//fica à escuta e bloqueia até que uma conexão seja estabelecida
BufferedReader in=new BufferedReader(new InputStreamReader(sock.getInputStream()))
BufferedWriter out = new BufferedWriter(new OutputStreamWriter(sock.getOutputStream()))
...
while(...){
    in.readLine();
    out.write(...);
    out.flush();
    }
```

read()
write()

close()

```
sock.shutdownInput();
sock.shutdownOutput();
sock.close();
```

- 1. Implemente um servidor que aceite a ligação de um cliente de cada vez, e que devolva ao cliente cada linha de texto que este lhe envie. Nota: pode testar o servidor desenvolvido recorrendo ao comando telnet.
  - exemplo: telnet ip porto
- 2.Implemente um cliente para o servidor de eco desenvolvido no exercício anterior.

- 3. Implemente um servidor que aceite a ligação de um cliente de cada vez. O servidor receberá de cada cliente, uma sequência de inteiros — pode optar tanto pelo formato binário como de texto – que terminará quando detectar a situação de end of file na stream de leitura do socket. No final da leitura dos valores inteiros, o servidor devolve ao cliente a soma correspondente.

- 4. Implemente um cliente para o servidor de soma desenvolvido no exercício anterior.

Aula 7: ex extra

- Modificar o servidor de eco do exercício 1 de modo a ser multi-threaded, ou seja para aceitar várias conexões simultâneas de diferentes clientes. Teste, por exemplo, com 2 conexões telnet ou usando o cliente eco.