PROCESSOS E COMUNICAÇÃO EM SISTEMAS DISTRIBUÍDOS

PROCESSO

Processo é definido como um programa em execução, e ***threads*** são fluxos ou linhas de execução dentro de um processo (TANENBAUM; BOS, 2015).

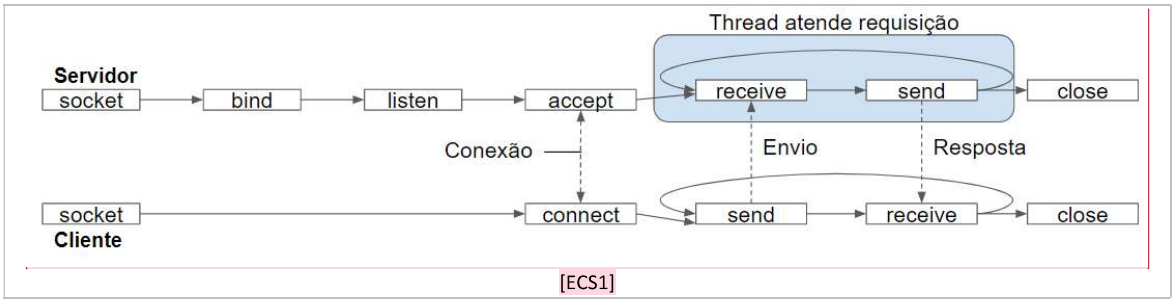
No contexto de comunicação entre um par de processos, o processo que inicia a comunicação é o **cliente**, e o processo que espera pela requisição é o **servidor** (KUROSE; ROSS, 2013).

Socket é uma interface de software que permite que um processo envie e receba mensagens da rede (KUROSE; ROSS, 2013).

É uma interface de programação da aplicação (*Application Programming Interface* – API) entre a aplicação e a rede. O programador escolhe o protocolo (TCP ou UDP) e utiliza os serviços desse protocolo para implementar a aplicação.

Padrão de comunicação orientada à comunicação com *sockets*

Comunicação entre um processo cliente e um processo servidor, na qual um *thread* no servidor atende à requisição do cliente.



Fonte: adaptada de Tanenbaum e Steen (2007, p. 86).

Esse *thread* é parte de um processo executado no servidor, que fica bloqueado em *receive* aguardando requisições do cliente, enquanto o processo principal retorna a primitiva *accept*, aguardando novas conexões.

O servidor poderia ter muitos *threads* executados paralelamente.

O sistema operacional é responsável pelo escalonamento de *threads* no uso dos recursos da máquina.

Saiba mais

Uma arquitetura na qual um processo pode funcionar como cliente e servidor ao mesmo tempo é do tipo Peer-to-Peer (P2P). A aplicação utiliza a comunicação direta entre pares de processos, sem passar por servidor dedicado. As aplicações de compartilhamento de arquivos (como BitTorrent), de telefonia por internet (como Skype) e de IPTV (como PPstream) são exemplos de arquiteturas P2P (KUROSE; ROSS, 2013).