Universidade Católica de Pelotas
Engenharia de Computação

API para protocolos internet:

Gabriel Harter Zoppo

Disciplina: Sistemas Distribuídos
Professora: Adenauer Correa Yamin

Pelotas, junho de 2022.

As características da comunicação entre processos

- **Comunicação síncrona:**
  - Os processos origem fazem as mensagens serem adicionadas em filas remotas.
  - Os processos destino removem mensagens de suas filas locais.
  - Processos origem e destino são sincronizados a cada mensagem, sendo que ambas operações causam bloqueio.
- **Comunicação Assíncrona:**
  - O uso da operação send é não bloqueante.
  - A transmissão da mensagem ocorre em paralelo com o processo origem.
  - A operação receive pode ter variantes com e sem bloqueio.

As características da comunicação entre processos

- **Destinos de mensagem:**
  - Uma porta local é um destino de mensagem dentro de um computador, especificado como um valor inteiro.
  - Qualquer processo que saiba o número de uma porta pode enviar uma mensagem para ela.
  - Se o cliente usa um endereço IP fixo para se referir a um serviço, então esse serviço sempre deve ser executado no mesmo computador para que seu endereço permaneça válido.

As características da comunicação entre processos

- **Confiabilidade:**
  - Um serviço de mensagem ponto a ponto pode ser descrito como confiável se houver garantia de que as mensagens foram entregues.
  - Um serviço de mensagem ponto a ponto pode ser descrito como não confiável se não houver garantia de entrega das mensagens.
  - Quanto à integridade, as mensagens devem chegar não corrompidas e sem duplicação

As características da comunicação entre processos

- **Ordenamento:**
  - Algumas aplicações exigem que as mensagens sejam entregues na ordem de emissão.
  - A entrega de mensagens fora da ordem da origem é considerada uma falha por tais aplicações.

Comunicação por datagrama UDP

- Para enviar ou receber mensagens, um processo precisa primeiro criar uma associação entre um soquete com um endereço IP e com uma porta do host local.
- Um servidor associa seu soquete a uma porta de serviço – que ele torna conhecida dos clientes para que estes possam enviar mensagens a ela. Um cliente vincula seu soquete a qualquer porta local livre.
- O método receive retorna, além da mensagem, o endereço IP e a porta da origem permitindo que o destinatário envie uma resposta a este.

Comunicação por datagrama UDP

- **Tamanho da mensagem:**
  - O processo destino precisa especificar um vetor de bytes de um tamanho em particular para receber as mensagens. Se a mensagem for grande demais para esse vetor, ela será truncada na chegada.
  - Qualquer aplicativo que exija mensagens maiores do que o tamanho máximo deve fragmentá-las em porções desse tamanho.

Comunicação por datagrama UDP

- **Bloqueio:**
  - Normalmente, os soquetes fornecem operações send não bloqueantes e receive bloqueantes para comunicação por datagrama (um receive não bloqueante é uma opção possível em algumas implementações).
  - A operação send retorna quando tiver repassado a mensagem para as camadas UDP e IP subjacentes, que são responsáveis por transmiti-la para seu destino.
  - Ao chegar, a mensagem é posta em uma fila de recepção vinculada ao soquete associado à porta de destino. A mensagem é recuperada dessa fila quando uma operação receive for realizada, ou estiver com sua execução pendente, nesse soquete.

Comunicação por datagrama UDP

- **Timeouts:**
  - Para evitar que um processo espere muito para receber alguma mensagem , limites temporais (timeouts) podem ser configurados nos soquetes.
  - O limite deve ser grande, em comparação com o tempo exigido para transmitir uma mensagem.

Comunicação por datagrama UDP

- **Recepção anônima:**
  - O método receive não especifica uma origem para as mensagens. A invocação ao método receive obtém uma mensagem endereçada para seu soquete, independentemente da origem.
  - O método receive retorna o endereço IP e a porta local do processo origem, permitindo que o destinatário verifique de onde ela veio. Entretanto, é possível associar um soquete de datagrama a uma porta remota e a um endereço IP em particular, no caso em que se deseje apenas enviar e receber mensagens desse endereço.

Comunicação por datagrama UDP

- **Modelo de falhas:**
  - A propriedade da integridade exige que as mensagens não devam estar corrompidas nem estejam duplicadas
  - **Falhas por omissão:** Mensagens podem ser descartadas devido a erros de soma de verificação ou porque não há espaço disponível no buffer, na origem ou no destino.
  - **Ordenamento:** Mensagens podem ser entregues em uma ordem diferente da que foram emitidas.
  - Um serviço de entrega confiável pode ser construído a partir de outro que sofra de falhas por omissão, pelo uso de confirmações.

Comunicação por fluxo TCP

- **Características da rede que são ocultas pela abstração de fluxo:**
  - **Tamanho das mensagens:** O aplicativo pode escolher o volume de dados que vai ser enviado ou recebido em um fluxo.
  - **Mensagens perdidas:** O protocolo TCP usa um esquema de confirmação, se não receber confirmação ele retransmite.
  - **Controle de fluxo:** O protocolo TCP tenta combinar a velocidade dos processos que leem e escrevem em um fluxo, se o processo envia muito rápido ele será bloqueado até a mensagem chegar.

Comunicação por fluxo TCP

- **Características da rede que são ocultas pela abstração de fluxo:**
  - **Duplicação e ordenamento de mensagens**: identificadores de mensagem são associados a cada datagrama IP, o que permite ao destinatário detectar e rejeitar duplicatas ou reordenar as mensagens que chegam fora da ordem de emissão.
  - **Destinos de mensagem:** Dois processos se conectam e podem se comunicar sem necessidade de usar endereços IP e portas.

Comunicação por fluxo TCP

- **Problemas importantes:**
  - **Correspondência de itens de dados:** Dois processos que estejam se comunicando precisam concordar quanto ao conteúdo dos dados transmitidos por um fluxo.
  - **Bloqueio:** Os dados gravados em um fluxo são mantidos em uma fila no soquete de destino, se um processo tentar ler dados de um canal de entrada, obterá dados da fila ou será bloqueado até que dados se tornem disponíveis.
  - **Threads:** Quando um servidor aceita uma conexão, ele geralmente cria uma nova thread para se comunicar com o novo cliente, evitando assim o travamento de outros clientes.

Comunicação por fluxo TCP

- **Modelo de falhas:**
  - Fluxos TCP usam somas de verificação para detectar e rejeitar pacotes corrompidos, assim como números de sequência para detectar e rejeitar pacotes duplicados.
  - Fluxos TCP usam timeout e retransmissões para lidar com pacotes perdidos.
  - Se a perda de pacotes em uma conexão ultrapassar um limite, será declarada que a conexão está desfeita.

Comunicação por fluxo TCP

- **Serviços que usam TCP:**
  - **HTTP:** O protocolo de transferência de hipertexto é usado para comunicação entre navegadores e servidores Web;
  - **FTP:** O protocolo de transferência de arquivos permite a navegação em diretórios em um computador remoto e que arquivos sejam transferidos de um computador para outro por meio de uma conexão.
  - **Telnet:** O serviço telnet dá acesso a um computador remoto por meio de uma sessão de terminal.
  - **SMTP:** O protocolo de transferência de correio eletrônico é usado para enviar correspondência entre computadores.

Universidade Católica de Pelotas
Engenharia de Computação

API para protocolos internet:

Gabriel Harter Zoppo

Disciplina: Sistemas Distribuídos
Professora: Adenauer Correa Yamin

Pelotas, junho de 2022.