Universidade Federal de São Paulo – UNIFESP
Instituto de Ciência e Tecnologia – ICT
Campus São José dos Campos

UC Sistemas Distribuídos

Prof. Bruno Kimura
15 de Maio de 2017

Metodologia de Trabalho: resolver a prática em grupos de 2 alunos.

Forma de entrega: Os códigos contendo a solução devem ser enviados pela plataforma EAD. No início do código insira comentário contendo os nomes dos integrantes.

Prática: Exclusão Mútua Distribuída

Implemente um protocolo para prover exclusão mútua a um conjunto de *N* clientes que desejam acessar um recurso comum entre eles. Esse protocolo deverá operar de acordo com o algoritmo d o ***Servidor Central***. Assuma que os clientes se comunicam com o servidor através de datagramas UDP.

Na implementação deste exercício, assuma que o recurso é uma variável do tipo inteiro que desempenha a tarefa simples de um **contador**. Quando um cliente obtém o *token* ele então poderá manipular o valor desse contador. Após o termino do acesso ao recurso, o cliente deverá notificar ao servidor que o recurso pode ser liberado, bem como o novo valor do contador.

Para tanto, utilize o formato de mensagens conforme a tabela abaixo.

Mensagem	**Descrição**
REQUEST_TOKEN = {PID_c, T_c}	O cliente que deseja acessar o recurso deverá enviar uma mensagem do tipo REQUEST_TOKEN ao servidor. Essa mensagem deverá conter o PID do cliente requisitante (PID_c) e seu tempo lógico (T_c).
REPLY_ALLOW = {PIC_c, T_s, recurso}	Ao conceder o *token* ao cliente, o servidor imediatamente deverá enviar a esse cliente uma mensagem do tipo REPLY_ALLOW notificando-o que o mesmo detém o acesso ao recurso. Essa mensagem deverá conter o PID desse cliente, o tempo lógico do servidor e o recurso (nesse caso, o valor da variável).
RELEASE_TOKEN = {PIC_c, T_c, recurso*}	Quando o cliente terminar de acessar o recurso ele deverá enviar ao servidor uma mensagem do tipo RELEASE_TOKEN, que contém seu PID_c, seu tempo lógico atual, bem o valor atualizado do recurso (recurso*).

A sincronização entre os nós do sistema distribuído deverá ser realizada conforme o algoritmo de ***Ordem Total***. Note que toda mensagem do protocolo contém o tempo lógico T do emissor e seu PID. Dessa forma, conforme as mensagens vão sendo trocadas entre cliente e servidor, ambos devem atualizar seus respectivos tempos lógicos, ou seja, T_atual = max(T_s, T_c) + 1.

O *token* é uma variável de controle do servidor para que o mesmo saiba que o recurso está em uso ou não, e se estiver, qual o processo está acessando a região crítica. Dessa forma, no servidor, o *token* pode ser representado por uma variável do tipo inteiro longo. O *token* deve ser inicializado com o valor -1, o que indicará que o recurso não está sendo acessado por nenhum processo.

Quando o servidor receber uma mensagem REQUEST_TOKEN:

- Se o recurso não estiver em uso, ou seja, se o valor *token* for -1 e a fila estiver vazia: o servidor então deverá atualizar o valor do *token* com o PID do cliente e notificar esse cliente com a mensagem REPLY_ALLOW.
- Se o recurso estiver em uso, ou seja, o valor do *token* for diferente de -1: o servidor deverá enfileirar a requisição do cliente de forma ordenada, conforme o tempo lógico T_c do cliente. Note que essa fila ordenada deverá priorizar clientes com menor tempo lógico, inserindo-os mais próximos do início da fila, de forma ordenada. Na prática, o efeito desse política de inserção na fila será o de penalizar clientes que solicitam o recurso frequentemente e priorizar clientes que solicitam pouco o recurso. Esse cliente requisitante então ficará aguardando a resposta do servidor REPLY_ALLOW.

Quando o servidor receber uma mensagem RELEASE_TOKEN:

- Se a fila estiver vazia, o servidor deverá atualizar o valor do *token* para -1, bem como o valor do recurso*, o qual foi manipulado pelo cliente.
- Se a fila não estiver vazia, o servidor deverá: atualizar o recurso conforme o novo valor manipulado pelo cliente, retirar da cabeça da fila a requisição pendente do próximo cliente, atualizar o valor do *token* conforme o PID do próximo cliente que irá acessar o recurso e, por fim, notificar esse cliente com a mensagem de REPLY_ALLOW.

Quando o cliente receber a mensagem REPLY_ALLOW:

- Ele deverá atualizar seu tempo lógico e, de forma arbitrária e por um tempo arbitrário, manipular o recurso. Tão logo terminar a manipulação do recurso, o cliente deverá enviar a mensagem de RELEASE_TOKEN ao servidor.

No servidor, a fila deverá ser implementada com inserção ordenada pelo tempo lógico do cliente solicitante. O elemento a ser enfileirado deve conter, por exemplo, a requisição do cliente (REQUEST_TOKEN) e seus dados de comunicação (struct sockaddr_in), de modo que o servidor consiga enviar a resposta REPLY_ALLOW ao cliente correto.