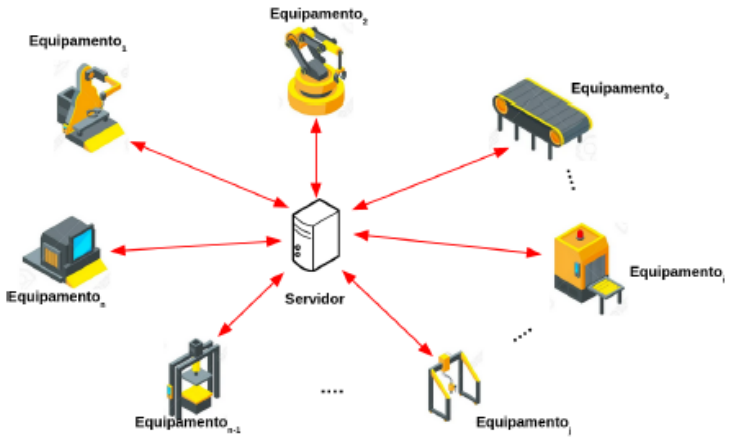
Documentação referente ao trabalho prático 2 da disciplina de Redes de Computadores
Rafael Antonio Armond Maia - 2019027148

Introdução

O trabalho prático visa a implementação de um sistema que funciona no modelo de cliente e servidor, onde o servidor pode estabelecer conexão com múltiplos clientes simultaneamente. A comunicação entre cliente e servidor é feita utilizando sockets. O servidor consegue se conectar a múltiplos clientes utilizando sockets e threads. O cliente utiliza apenas uma thread, um fluxo, o fluxo principal do programa, espera por entradas do teclado enquanto que a thread espera por mensagens do servidor.

Desenho arquitetural geral do sistema

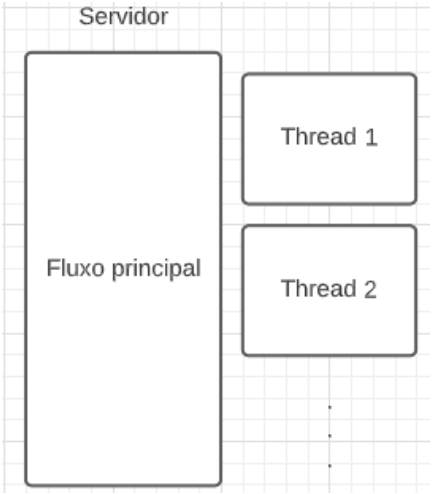


O fluxo da thread do cliente também é responsável por processar as informações que vêm do cliente. As informações trafegam entre os nós utilizando código predeterminados, quando o código chega no cliente ou no servidor, este se encarrega de processar a informação conforme o código recebido. A maior parte dos códigos que trafega na rede vai com uma carga útil, onde o destinatário utiliza dessa carga no processamento da informação e resulta na exibição da informação correta no console.

Arquitetura

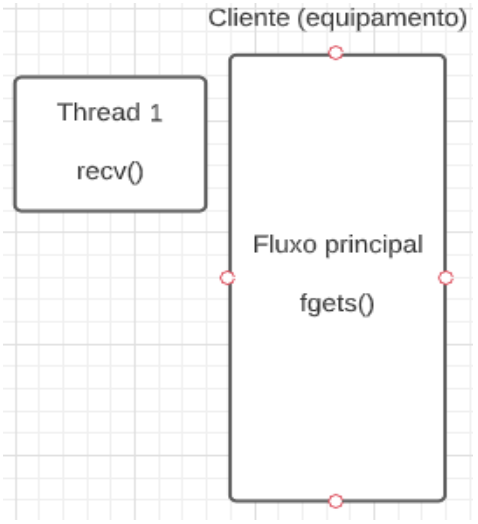
Começando pela arquitetura do servidor, este tem a tarefa primordial de se conectar a múltiplos clientes simultaneamente. Para tal, o servidor utiliza múltiplos threads, cada vez que um cliente solicita estabelecimento de conexão com o servidor é feita uma verificação para checar se o número máximo de clientes não foi atingido e caso não tenha sido, uma nova thread é criada para “conversar” com o cliente.

Desenho arquitetural do servidor

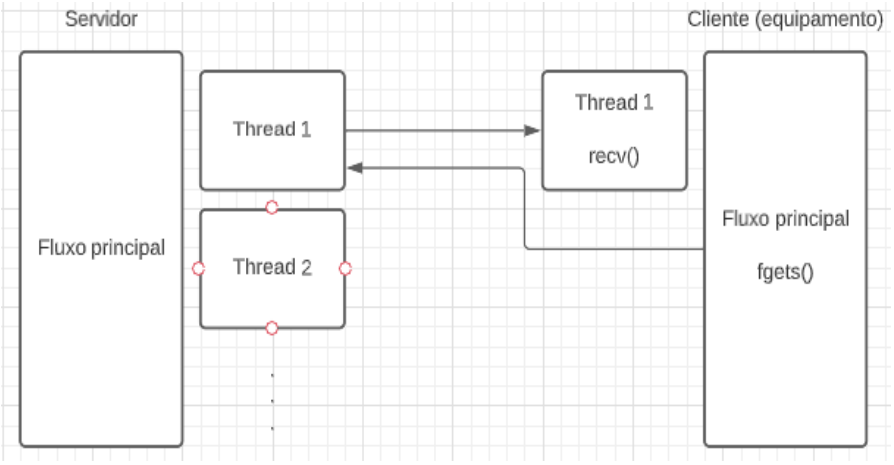


O cliente possui uma arquitetura para paralelizar a chegada de dados do servidor e o input do usuário pelo teclado. Cria-se uma única thread para aguardar dados do servidor enquanto o fluxo principal é responsável por aguardar dados do usuário e enviá-los ao servidor quando o usuário terminar o input de dados.

Desenho arquitetural do cliente (equipamento)



Desenho da comunicação entre o cliente e servidor



Servidor

O servidor é responsável por aceitar a conexão com o cliente, fechar a conexão com o cliente e fornecer informações e serviços ao cliente, conforme este solicite ao servidor.

O servidor tem três variáveis de escopo global que são muito importantes para seu funcionamento correto. São elas: o vetor de equipamentos, do tipo struct client_data, e que tem um limite máximo de 15 posições, pois este é o número máximo de equipamentos que podem se conectar simultaneamente ao servidor. E a variável “numEquipmentsConnected”, do tipo inteiro, que representa o tamanho atual do vetor de equipamentos conectados. Por último, a variável do tipo inteiro “equipmentId”, que guarda o número que será atribuído ao identificador do equipamento que se conecta ao servidor.

O servidor recebe códigos e usa uma estrutura de controle “switch” para determinar o que fazer com cada código. Cada case do switch processa o código recebido de acordo com sua finalidade.

Códigos recebidos pelo servidor:

- Código 1 - Inserir novos equipamentos/clientes: O servidor chama uma função para inserir novos clientes e passa as informações do cliente como parâmetro. A função

“insert_new_equipment(void *data) é a função responsável por adicionar um novo equipamento ao vetor de equipamentos do servidor e atualizar a variável que representa o número de equipamentos conectados. Essa função também envia para o cliente que acabou de se conectar, o seu identificador (ID) e envia para todos os outros equipamentos conectados ao servidor uma mensagem informando que um novo equipamento se conectou e seu identificador. As mensagens seguem o formato estabelecido na descrição do TP.

- Código 2 - Desconectar um equipamento do servidor: O servidor chama uma função para desconectar um equipamento que está armazenado no vetor de equipamentos do servidor. Esta função é a “int remove_equipment(int equipment_id)”. Essa função procura o equipamento no vetor de equipamentos através do ID passado por parâmetro, remove o elemento e envia a mensagem correta para todos os outros equipamentos do vetor, que estão conectados ao servidor, informando que o equipamento X foi desconectado.
- Código 3 - Trata a requisição de informação de um equipamento especificado pelo seu identificador: Envia uma mensagem “requested information” como payload de uma mensagem para o equipamento requerido. O equipamento requerido imprime no seu console o payload. O servidor envia para o equipamento requerente um valor aleatório, com duas casas decimais, variando entre 1 e 10, como payload da mensagem enviada ao equipamento requerente.
- Código 4 - Trata requisição de listagem de equipamentos conectados ao servidor: O servidor percorre o vetor de equipamentos e cria um payload com todos os identificadores dos equipamentos conectados à rede. Vale lembrar que há uma lógica para excluir o ID do equipamento requerente. Após a montagem do payload o servidor envia de volta para o cliente uma mensagem contendo um código que será processado pelo cliente e o payload, que é uma string contendo todos os IDs dos equipamentos conectados ao servidor.
- Caso o código identificado pelo servidor não seja nenhum dos listados acima, o servidor imprime no console uma mensagem informando que o comando não foi reconhecido.

Cliente

O cliente é responsável por capturar as entradas do teclado informadas pelo usuário e por receber mensagens do servidor.

O cliente possui duas variáveis de escopo global importantes para seu funcionamento correto, são elas: “equipment_id”, que é uma variável de tipo inteiro, e que tem por finalidade guardar o identificador do equipamento e “target_equip_id”, que é uma variável de tipo inteiro, e que tem por finalidade guardar o ID do equipamento requerido.

O cliente recebe códigos do servidor e usa uma estrutura de controle “switch” para determinar o que fazer com cada código. Cada case do switch processa o código recebido de acordo com sua finalidade.

Códigos recebidos pelo cliente:

- Código 1 - Trata a confirmação de conexão do equipamento corrente ou da conexão de algum outro equipamento: Ao identificar o código 1, o cliente verifica se seu ID já foi atribuído, caso não tenha sido, imprime no console a mensagem “New ID: <X>”. Caso seu ID já tenha sido atribuído, ele reconhece que é o caso de um novo equipamento estar se conectando ao servidor e imprime a mensagem especificada na descrição do TP.
- Código 2 - Trata a confirmação de desconexão do equipamento corrente ou da desconexão de algum outro equipamento: Ao identificar o código 2, o cliente verifica se o identificador do equipamento é o seu, se for o caso, exibe a mensagem especificada e interrompe o fluxo do programa, caso não seja o caso, imprime a mensagem especificada informando que outro equipamento interrompeu sua conexão com o servidor.
- Código 3 - Trata a mensagem contendo a listagem de identificadores: Ao identificar o código 3, o cliente pega o payload da mensagem enviada pelo servidor, que é uma string contendo os identificadores dos equipamentos conectados e exibe em tela, como no exemplo: “01 02 03”.

- Código 4 - Trata a mensagem contendo o valor de leitura de um equipamento X conectado ao servidor, que não é o equipamento corrente: Ao identificar o código 4, o cliente separa o payload, que é uma string contendo o valor de leitura de um equipamento X, conectado ao servidor e monta uma mensagem para imprimir no console. Por exemplo: “Value from 02: 8.21”, onde 02 é o equipamento que foi requerido pelo equipamento corrente e 8.21 é o valor lido desse equipamento.
- Código 5 - Trata a mensagem contendo um aviso para o equipamento: Ao identificar o código 5, o cliente imprime no console o payload da mensagem, que é uma string informando que outro equipamento da rede solicitou dados do equipamento corrente. A mensagem é “requested information”, especificada pelo enunciado do TP.
- Código 6 - Trata a mensagem informando que o equipamento requerido pelo equipamento corrente não está na rede: Ao identificar o código 6, o equipamento imprime a mensagem no console informando que o equipamento requerido por ele não está conectado ao servidor.

Discussão

O desenvolvimento do código foi dividido em etapas, como sugerido na descrição do TP. A primeira tarefa foi estabelecer uma conexão múltipla com vários clientes, e posteriormente o envio e tratamento das mensagens enviadas tanto pelo servidor quanto pelo cliente. O desenvolvimento do cliente foi particularmente difícil até que houve a ideia de paralelizar o fluxo principal com uma thread. Sendo assim, o fluxo principal fica esperando o usuário informar dados através do teclado, e a thread espera mensagens vindas do servidor. Sendo assim, é possível tanto enviar mensagens vindas do fgets() para o servidor, quanto esperar mensagens do servidor.

Conclusão

O sistema atende a todas as especificações estabelecidas na descrição do trabalho prático dois. Foram realizados diversos testes, inclusive o exemplo contido no final da especificação do enunciado do trabalho prático.