Cliente/Servidor GDBM usando Sockets TCP e UDP

Alessandra Santos[1], Lorrayne F. Freitas[1], Yuri C. Ferreira da Silva[1]
[1]Bacharelado em Sistemas de Informação – Faculdade de Computação
Universidade Federal de Uberlândia (UFU)
Av. João Naves de Ávila, 2121 – Uberlândia – MG
{alesantos_qi@yahoo.com.br, lorrayne.florencio, yuricampos@gmail.com}

Resumo

Este projeto tem como objetivo explicitar os conceitos de uma comunicação entre clientes e servidores, realizar o desenvolvimento da aplicação prática destes conceitos juntamente com a tecnologia necessária e, consequentemente, tornar a compreensão sobre o assunto algo concreto e válido.

Para que essa comunicação fosse válida, fez-se necessário seguir e respeitar algumas restrições. Para se enquadrar dentro destas restrições, a comunicação do cliente/servidor foi elaborada via biblioteca de Sockets TCP e UDP. A aplicação distribuída também foi desenvolvida sobre a base de dados GDBM (*GNU DataBase Manager*) disponível no sistema operacional UNIX. Ademais, o servidor deverá ter a competência de acolher as requisições do cliente enviadas pela rede, e posteriormente executá-las de forma eficiente. Prontamente, o cliente é o encarregado pela realização da interface com o usuário.

Para a comunicação do cliente/servidor se realizar de forma correta é necessário, esse processamento deve ser via biblioteca de *sockets* UDP e TCP. Sendo que o *socket* é a interface que permite conversar com outros programas, representando um extremo de uma conexão, utilizando o protocolo de transporte UDP (*User Datagram Protocol* - Protocolo de Datagrama do Usuário) ou TCP (*Transmission Control Protocol* - Protocolo de Controle de Transmissão).

A aplicação desenvolvida, observando as restrições citadas e seguindo as etapas sequenciais, oferece a funcionalidade de armazenamento de nomes e números de telefones, sendo uma agenda eletrônica não complexa. De forma consolidada e íntegra o programa oferece os serviços de registro ou atualização de um contato, a exclusão ou a consulta de um contato e, por fim, a finalização da aplicação. Com isso, o cliente deverá ser notificado com uma resposta ao final de cada execução de uma requisição solicitada.

Introdução

Neste projeto dois tipos de *sockets* são abordados: UDP (*User Datagram Protocol* - Protocolo de Datagrama do Usuário) e TCP (*Transmission Control Protocol* - Protocolo de Controle de Transmissão). No que diz respeito ao desenvolvimento do projeto, as características dos protocolos de transporte TCP e UDP devem ser respeitadas.

O protocolo de transporte TCP é orientado a conexão, tem controle de erro, garante a entrega, e a mesma é de forma ordenada. O protocolo UDP não necessita de conexão, não possui controle de erro, e é mais rápido e simples.

As características da comunicação entre cliente e servidor utilizando *sockets* TCP são: servidor fica à espera de ligações; cliente liga-se ao servidor (conexão estabelecida); conexão é um canal fiável para comunicação bidirecional; um socket representa um extremo de uma conexão; uma conexão é caracterizada por um *socket*-

pair. As características de uma conexão TCP são: a fiabilidade, ordem, controle de fluxo e *full-duplex* (envia e recebe dados). No endereçamento, cada máquina (host) tem (pelo menos) um endereço IP; podem existir vários serviços em cada máquina; serviços de uma máquina são distinguidos por portos (um servidor especifica em que porto escuta); um cliente especifica onde se ligar com um par.

O protocolo UDP não é do tipo confiável; como é um serviço sem conexão, logo um cliente UDP pode criar um *socke*t, enviar um datagrama para um servidor e enviar outro datagrama com o mesmo *socket* para um tipo diferente de servidor.

O projeto em questão concerne no desenvolvimento de uma aplicação distribuída de base dados disponível no UNIX (GDBM). A aplicação é uma agenda eletrônica que armazena nomes e números de telefones. A comunicação entre cliente e servidor deverá se processar via biblioteca de *sockets* TCP e UDP. O programa apresenta um menu na seguinte ordem: 1 - Armazena/Atualiza um Registro; 2 - Remove um Registro; 3 - Acessa um Registro; 4 - Finaliza a Aplicação. Se o cliente solicita a opção 1, o programa pede o nome e o número de telefone. Se já existir registro, o programa retorna uma alerta ao usuário pedindo permissão para atualizar registro. Opções 2 e 3 apenas o nome é solicitado. A opção 2 retorna mensagem de sucesso ou falha. Na opção 3, o programa retorna o número do telefone ou um indicativo de falha. E a Opção 4, como o próprio nome sugere, finaliza a aplicação.

Seções Específicas

Para a implementação da agenda eletrônica utilizando comunicação entre cliente e servidor processada via Biblioteca de *sockets* TCP e UDP, foram necessários à implementação de funções que enviam dados no TCP e UDP, além das funções que gravam, atualizam e deletam no banco de dados, utilizando o GDBM (disponível no UNIX).

server TCP:

recv(ns, &c, sizeof(c), 0); recebe do cliente
send(ns, &c, sizeof(c), 0); envia para o cliente

client TCP:

send(s, &c, sizeof(c), 0); envia para o servidor
recv(s, &c, sizeof(c), 0); recebe do servidor

client UDP:

sendto(s, &c, sizeof(c), 0, (struct sockaddr *)&server, sizeof(server)); envia para o servidor
read(s,&c,sizeof(c)); recebe do servidor

server UDP:

recvfrom(s, &c, sizeof(c), 0, (struct sockaddr *) &client, &client_address_size); recebe do cliente
sendto(s,&c,sizeof(c),0,(struct sockaddr *)&client,client_address_size); envia para o cliente

Banco de Dados:

gdbm_store - novo registro
gdbm_delete - apaga registro
gdbm_fetch - obtém registro

gdbm_store(dbf, key, data, GDBM_INSERT); armazena informações no banco de dados

gdbm_delete(dbf, key); apaga do banco de dados

gdbm_fetch(dbf, key); obtém o dado

gdbm_close(dbf); fecha banco de dados

Resultados

Os resultados obtidos a partir da execução da aplicação com todas as requisições respeitadas foram satisfatórios, englobando assuntos dentro do previsto e desejáveis.

De uma maneira bastante sucinta e objetiva é apresentado, abaixo em forma de tabelas, os resultados coletados, abordando a comunicação do cliente/servidor e a troca de mensagens entre estes, utilizando os protocolos de comunicação TCP e UDP.

Os resultados da visão do lado do Servidor TCP:

Servidor TCP	
Iniciar	Informando a porta de conexão como parâmetro o sistema foi iniciado com sucesso.
Inserir / Atualizar	Utilizando dados recebidos do cliente e, posteriormente, a atualização foi salva com sucesso através dos novos dados, sendo eviada uma resposta para o cliente, o status.
Remover	Utilizando o nome do contato (sendo que esta informação é a chave de acesso para o banco de dados) do qual o cliente enviou ao servidor, o registro foi removido do banco de dados com sucesso, assim o cliente recebeu a mensagem de êxito. Quando o registro não existe no banco, uma mensagem do status é enviada ao cliente informando que o contato não existe.
Acessar Registro	Utilizando o nome do contato recebido do cliente, verifica no banco de dados se existe este contato, em caso afirmativo, foi retornado ao cliente os dados que foram solicitados. Caso o contato não existe no banco de dados é enviado ao cliente o status informando que o contato não existe.
Finalizar	Caso existe mais de um cliente no servidor, então o aplicativo não será encerrado, caso contrário o aplicativo é encerrado com sucesso.

Fig. 01 – Resultados obtidos a partir do Servidor TCP.

Os resultados da visão do lado do Cliente TCP:

Cliente TCP	
Iniciar	Informando a porta de conexão como parâmetro e o endereço do servidor a conexão foi realizada com sucesso.
Inserir / Atualizar	Informando o nome e telefone a serem inseridos/atualizados, estes dados são enviados para o servidor e foi recebida a resposta de sucesso da inserção ou da atualização, se caso o registro já existisse no banco.
Remover	Informando o nome do contato a ser removido, sendo essas informações enviadas ao servidor, foi recebida a resposta de sucesso ou não por parte do servidor.
Acessar Registro	Informando o nome, enviou ao servidor, foi recebido o status de sucesso e os dados do contato solicitado, se caso existiam no banco de dados. Caso contrário, é recebido o status de insucesso.
Finalizar	Escolhida esta solicitação, o cliente foi encerrado.

Fig. 02 – Resultados obtidos a partir do Cliente TCP.

Os resultados da visão do lado do Servidor UDP:

Servidor UDP	
Iniciar	Informando a porta como parâmetro o sistema foi iniciado com sucesso.
Inserir / Atualizar	Utilizando dados recebidos do cliente e, posteriormente, a atualização foi salva com sucesso através dos novos dados, sendo eviada uma resposta para o cliente, o status.
Remover	Utilizando o nome do contato (sendo que esta informação é a chave de acesso para o banco de dados) do qual o cliente enviou ao servidor, o registro foi removido do banco de dados com sucesso, assim o cliente recebeu a mensagem de êxito. Quando o registro não existe no banco, uma mensagem do status é enviada ao cliente informando que o contato não existe.
Acessar Registro	Utilizando o nome do contato recebido do cliente, verifica no banco de dados se existe este contato, em caso afirmativo, foi retornado ao cliente os dados que foram solicitados. Caso o contato não existe no banco de dados é enviado ao cliente o status informando que o contato não existe.
Finalizar	Apesar da solicitação, não encerra servidor UDP.

Fig. 03 – Resultados obtidos a partir do Servidor UDP.

Os resultados da visão do lado do Cliente UDP:

Cliente UDP	
Iniciar	Informando a porta de conexão como parâmetro e o endereço do servidor a conexão foi realizada com sucesso.
Inserir / Atualizar	Informando o nome e telefone a serem inseridos/atualizados, estes dados são enviados para o servidor e foi recebida a resposta de sucesso da inserção ou da atualização, se caso o registro já existisse no banco.
Remover	Informando o nome do contato a ser removido, sendo essas informações enviadas ao servidor, foi recebida a resposta de sucesso ou não por parte do servidor.
Acessar Registro	Informando o nome, enviou ao servidor, foi recebido o status de sucesso e os dados do contato solicitado, se caso existiam no banco de dados. Caso contrário, é recebido o status de insucesso.
Finalizar	Escolhida esta solicitação, o cliente foi encerrado.

Fig. 04 – Resultados obtidos a partir do Cliente UDP.

Análise e Resultados da Implementação

Os resultados apresentados anteriormente foram satisfatórios, pois a aplicação se portou dentro dos padrões especificados.

A aplicação cotém o menu de opções para o cliente conforme foi pedido. Além disso, ela insere nome (chave de acesso) e telefone, atualiza apenas telefone, remove nome e telefone e acessa o registro verificando se o dado informado já existe no banco (apresentados na sessão de Resultados). Tudo via comunicação entre cliente e servidor processado via Biblioteca de *sockets* TCP e UDP. Com isso, foram desenvolvidos: um protocolo de comunicação cliente/servidor capaz de requisitar e receber o resultado de todas as operações "dbm" (disponível no UNIX (GDBM - Gnu DataBase Manager)); uma biblioteca de funções "dbm" que será utilizada pelo cliente que conterá todas as chamadas "dbm", direcionando-as ao servidor; implementação um servidor "dbm" que receba e processe requisições via protocolo desenvolvido no primeiro item e o teste da versão distribuída em uma Rede LAN (utilizando mais de um cliente).

Conclusão

Na comunicação orientada à conexão via TCP o servidor fica à espera de conexões, assim o cliente liga-se ao servidor, estabelecendo uma nova conexão. Sendo esta conexão uma comunicação bidirecional. As extremidades da conexão são representadas por um *socket*, sendo então caracterizado como um *socket-pair*. Segundo o autor James F. Kurose "o *socket* é a interface entre a camada de aplicação e a de transporte dentro de uma máquina". As principais características fundamentais do TCP é a orientação à conexão, sendo estabelecida entre dois pontos, utilizando esta conexão para transferir os dados, confiabilidade, utilizando várias técnicas para uma entrega dos dados da maneira mais correta possível, a transferência se dá em ambas direções, e

possui um controle de fluxo, onde o receptor envia mensagens como confirmação de recebimento de um segmento de dados. As várias qualidades e utilidades da conexão TCP deu-se por consequência a caracterização deste protocolo como o mais adequado para redes globais. Dentre estas podemos citar, resumidamente, os aspectos e características que de verificação dos dados para o envio destes sempre na sequência correta, de forma apropriada e sem erros, pela rede. Assim, para o envio de dados de modo confiável, é recomendável o uso de protocolo TCP.

Em contrapartida, o UDP é um protocolo simples, permitindo que uma aplicação escreva um datagrama, o encapsulo em um pacote e, só então, é enviado ao destino, não tendo nenhuma garantia que o pacote será recebido, nem mesmo se será na mesma sequência enviada. Com isso, é possível afirmar que este protocolo não é recomendável para utilizá-lo quando o que está sendo enviado tenha que ser entregue corretamente, assim, é um protocolo não é confiável. Em contrapartida, é um protocolo mais rápido e menos complexo. É categorizado como um serviço sem conexão, pois não há necessidade desta funcionalidade. Não perdendo tempo com a criação e destruição de conexão, torna-se mais rápido, como já citado. Para uso do UDP adequadamente e sem prejuízos é indicado para fluxos de dados em tempo real, para aqueles que permitem perda ou corrompimento de parte de seu conteúdo sem maiores dados, como transmissão de vídeos ou som.

Por conseguinte, o projeto apresentado reforça a utilização prática destes conceitos de maneira correta para a maior compreensão e conhecimento aprofundado. Para chegar aos resultados satisfatórios alcançados ao longo do desenvolvimento do projeto foi preciso o estudo teórico das definições e a força prática, tendo como consequência a absorção do conteúdo como um todo de modo mais claro e dinâmico.