DCC023 – Redes de computadores

Trabalho prático 1 – IRC Seletivo

Professores: Daniel Fernandes Macedo, Marcos Augusto M. Vieira

Monitor: Lucas Augusto Maia da Silva

Data de entrega: 04/11/2015

Trabalho em grupos de dois alunos (ou individual).

Valor do trabalho: 20 pontos

1. Introdução

Na Internet, um dos primeiros sistemas de bate-papo foi o IRC (Internet Relay Chat). O IRC suporta conversas por texto, em canais. Como todo lugar em que temos muitas pessoas, sempre existe alguém que está postando comentários que julgamos chatos ou irrelevantes, mas que outros acham interessante.

Baseado nessa ideia, foi proposto o sistema de IRC-2 com o comando MUTE. O MUTE trata exatamente dos casos em que o usuário não está interessado mais em saber o que uma outra pessoa está comentando. O IRC-2 é um protocolo que roda em cima do TCP. Ele possui comandos básicos, para definir o apelido do usuário, para enviar mensagens, e para tirar ou colocar um usuário em mudo. Para economizar banda dos clientes, é o próprio servidor que implementa a função de mudo, ou seja, ele não envia para o usuário as mensagens que ele não deseja ver (ao contrário de uma implementação em que o servidor envia todas as mensagens, mas o cliente é que filtra as mensagens indesejadas).

2. Programas a serem desenvolvidos

O trabalho prático consistirá na implementação de dois programas, o cliente e o servidor. Ambos vão receber parâmetros de entrada como explicado a seguir:

```
cliente <ip/nome> <porta> <usuario>
```

```
servidor <porta>
```

O primeiro programa, o cliente, irá conectar no servidor definido pelo endereço IP (ou nome, por exemplo *login.dcc.ufmg.br*) e porta passados por parâmetro. O terceiro parâmetro será uma *string*, que é o apelido do usuário no IRC-2.

Já o servidor irá executar um serviço, que necessariamente irá tratar várias conexões ao mesmo tempo, por se tratar de um servidor de *chat*. A porta a ser escutada é dada pelo parâmetro único do programa.

3. Protocolo IRC-2

Ambos o cliente e servidor irão se comunicar usando o protocolo IRC-2. O IRC-2 é baseado em texto, ou seja, os comandos são strings de caracteres ASCII. Além disso, todos os comandos são digitados com caracteres maiúsculos. Os seguintes comandos são suportados:

- **NICK <string>:** Define o nome de usuário que será utilizado pelo cliente durante uma conexão. Este deve ser o primeiro comando enviado pelo usuário. O apelido não deve conter caracteres de espaço, e tem um tamanho limitado a 16 caracteres.
- **POST <string>**: É empregado para enviar mensagens no sentido cliente → servidor. As mensagens são texto livre, de no máximo 500 caracteres, sempre terminados por um caractere "\n", que é contabilizado dentro do limite de 500 caracteres.
- **NEW <user> <string>**: É empregado para enviar mensagens no sentido servidor → cliente. As mensagens são texto livre, de no máximo 500 caracteres, sempre terminados por um caractere "\n", que é contabilizado dentro do limite de 500 caracteres. A string **<user>** identifica o usuário que enviou a mensagem.
- **MUTE <user>**: Empregado no sentido cliente → servidor, indica ao servidor que ele deve parar de enviar as mensagens de um dado cliente, identificado pelo apelido **<user>**, para o usuário.
- **UNMUTE <user>**: Empregado no sentido servidor → cliente, indica ao servidor que ele deve voltar a enviar as mensagens de um dado cliente, identificado pelo apelido **<user>**, para o usuário.
- **CLOSE**: Comando enviado pelo cliente, termina a conexão ao servidor. Quando o cliente enviar o CLOSE o servidor deverá limpar da sua memória a lista de usuários em mudo. Assimm quando o usuário logar novamente, ele irá receber as mensagens de todos os usuários, mesmo se ele os colocou em mudo em uma conexão anterior.

Vale ressaltar que todos os comandos são terminados por um "\n", e que os parâmetros de um comando são sempre separados por espaços (pode ocorrer mais de um caractere de espaço, mas outros caracteres de separação tais como tab ou '\r' não deverão ser considerados).

4. Entrega do código

O código e a documentação devem ser entregues em um arquivo Zip (não pode ser RAR nem .tgz nem .tar.gz) no Moodle, contendo um arquivo **readme.txt** com o nome dos integrantes, e um arquivo PDF da documentação. Incluam todos os arquivos (.c, .h, makefile, não incluam executáveis ou arquivos objeto) EM UM ÚNICO DIRETÓRIO. Um makefile deve ser fornecido para a compilação do código.

Parte desse trabalho envolve o aprendizado de como construir um makefile e utilizar a ferramenta Make. Este makefile, quando executado sem parâmetros, irá

gerar os dois programas, *cliente* e *servidor*, EXATAMENTE com esses nomes. Submissões onde os programas não seguem as especificações de parâmetros e nomes, makefiles não funcionando ou arquivos necessários faltando não serão corrigidos. Programas que não compilarem também não serão corrigidos. O julgamento do trabalho será feito em um computador rodando o SO Linux.

5. Documentação

A documentação, além de ser entregue com o Zip junto ao código, deve ser entregue impressa ao professor até o dia seguinte à entrega, na sala de aula. Caso não tenhamos aula no dia seguinte ao prazo para envio, iremos combinar antecipadamente qual é a forma mais conveniente de entrega da documentação. Trabalhos que forem entregues no Moodle mas sem a documentação impressa **não serão** corrigidos.

O texto da documentação deve ser breve, de forma que o corretor possa entender o que foi feito no código sem ter que entender linha a linha dos arquivos. Implementações modularizadas deverão mencionar quais funções são implementadas em cada módulo ou classe. A documentação deve conter os seguintes itens:

- Sumário do problema a ser tratado
- Uma descrição sucinta dos algoritmos e dos tipos abstratos de dados, das principais funções, e procedimentos e as decisões de implementação
- Decisões de implementação que porventura estejam omissos na especificação
- Testes, mostrando que o programa está funcionando de acordo com a especificação, seguidos da sua análise
- Conclusão e referências bibliográficas

6. Avaliação

A avaliação do trabalho será composta pela execução dos programas desenvolvidos e pela análise da documentação. Os seguintes itens serão avaliados:

- A qualidade do código (código bem organizado, com comentários explicativos, variáveis com nomes intuitivos, modularidade, etc).
- Execução correta do código em entradas de testes, a serem definidas no momento da avaliação. As entradas de teste irão exercitar a funcionalidade completa do código e testar casos especiais ou de maior dificuldade de implementação, mas que devem ser tratados por um programa correto.
- Aderência ao protocolo especificado. Iremos usar ferramentas de captura do tráfego da rede (*tcpdump*, *wireshark*) e iremos analisar o código para

verificar se a comunicação está implementada da forma definida na documentação.

- Conteúdo da documentação, que deve conter os itens mencionados anteriormente.

- Coerência e coesão da documentação (apresentação visual e organização, uso correto da língua, qualidade textual e facilidade de compreensão).

Como mencionado anteriormente, trabalhos fora da especificação (por exemplo, sem makefile, que não gerem programas com os nomes especificados, que não compilem ou não possuam os parâmetros esperados) não serão corrigidos. Trabalhos fora da especificação tomam muito trabalho do corretor, que deve entender como compilar e rodar **cada programa avaliado**, tempo esse que deveria ser empregado para avaliar a execução e corretude do código.

Qualquer elemento que não esteja especificado neste documento, mas que tenha que ser inserido para que o protocolo funcione, deve ser descrito na documentação de forma explícita.

7. Desconto de nota por atraso

Os trabalhos poderão ser entregues até a meia-noite do dia especificado para a entrega. O horário de entrega deve respeitar o relógio do sistema Moodle, ou seja, a partir de 00:01 do dia seguinte à entrega no relógio do Moodle, os trabalhos já estarão sujeitos a penalidades.

A fórmula para desconto por atraso na entrega do trabalho prático ou resenha é:

Desconto = $2^{d-1}/0.32$ %

onde *d* é o atraso em dias úteis. Note que após 5 dias úteis, o trabalho não pode ser mais entregue.

8. Referências

Programação em C:

- http://www.dcc.ufla.br/~giacomin/Textos/tutorialc.pdf
- ftp://ftp.unicamp.br/pub/apoio/treinamentos/linguagens/c.pdf
- http://www.cs.cf.ac.uk/Dave/C/CE.html
- http://www2.its.strath.ac.uk/courses/c/
- http://www.lysator.liu.se/c/bwk-tutor.html

Uso do programa make e escrita de Makefiles:

- http://comp.ist.utl.pt/ec-aed/PDFs/make.pdf
- http://haxent.com.br/people/ruda/make.html
- http://informatica.hsw.uol.com.br/programacao-em-c16.htm

- http://www.gnu.org/software/make/manual/make.html
- http://www.cs.colby.edu/maxwell/courses/tutorials/maketutor/