Rede de Computadores

Turma Grupo

up201806250@fe.up.pt Diogo Samuel Gonçalves Fernandes
up201806505@fe.up.pt Paulo Jorge Salgado Marinho Ribeiro

6 de novembro de 2020

Projeto RCOM - 2019/20 - MIEIC

Professor das Aulas Práticas:

Informação sobre o relatório do 1º trabalho laboratorial de RCOM

a) O relatório do 1º trabalho laboratorial de RCOM deverá conter a seguinte informação:

- Título, Autores

- Sumário (dois parágrafos: um sobre o contexto do trabalho; outro sobre as principais conclusões do relatório)

1 Introdução

Tendo como principal objetivo implementar um protocolo de transferência de dados recorrendo a uma porta série, este trabalho deve resultar num programa capaz de resistir a fenómenos como a interrupção da porta série ou a receção de informação corrompida, provocada pela indução de "ruído" na porta série. Este relatório procura explicar toda a teoria envolvida neste primeiro trabalho, de forma bem estrutura, nos seguintes tópicos: - Arquitetura Descrição dos blocos funcionais e interfaces - Estrutura do código Explicação das APIs, enumeração das principais estruturas de dados utilizadas, das funções de maior importância e relação com a arquitetura - Casos de Uso Principais Identificação dos casos de uso mais importantes, e demonstração sequencial das chamadas às funções. - Protocolo de ligação lógica Identificação dos principais aspetos funcionais da camada de Ligação de Dados, descrição da estratégia de implementação destes aspetos, com o apoio de extratos de código. - Protocolo de aplicação Identificação dos principais aspetos funcionais da camada da Aplicação, descrição da estratégia de implementação destes aspetos, com o apoio de extratos de código. - Validação Descrição dos testes efetuados com apresentação quantificada dos resultados. - Eficiência do protocolo de ligação de dados Caracterização estatística da eficiência do protocolo, feita com recurso a medidas sobre o código desenvolvido. - Conclusões Síntese da informação apresentada nas secções anteriores, e reflexão sobre os objetivos de aprendizagem alcançados.

2 Arquitetura

O programa desenvolvido desdobra-se em duas camadas bem definidas: a camada de Ligação de Dados (Link Layer) e a camada da Aplicação (Application Layer). A primeira, como é responsável pelo estabelecimento da ligação, torna o protocolo sólido, garantindo a sua consistência. Por este motivo, é considerada a camada de mais baixo nível do programa, sendo que trata da abertura da porta série, da transmissão de informação (escrita e leitura), e do seu posterior fecho. É também da sua responsabilidade testar se a informação foi escrita/recebida corretamente, através de byte stuffing, e de testes de erros como os BCC, que serão aqui detalhados mais tarde.Por outro lado, a camada da Aplicação é apenas responsável pelo envio e receção da informação dos ficheiros, pelo que é de um nível superior à camada de ligação de dados. Assim, esta camada chama as funções da camada da ligação de dados, para envio/receção da informação de dados, mantendo-se, no entanto, completamente independente desta, uma vez que desconhece os seus métodos de atuar.

3 Estrutura do código

O Makefile por nós criado efetua a compilação do programa, resultando em dois executáveis diferentes, um para o Emissor (writenoncanonical) e outro para o Recetor (noncanonical). Emissor: O executável relativo ao Emissor exige 2 argumentos: o nome da porta série (por exemplo /dev/ttyS1), e o nome do ficheiro que vai ser transmitido (exemplo: pinguim.gif) A sequência de chamadas efetuada por este executável é a seguinte: - llopen: Configura a ligação entre os dois computadores, abrindo a porta série em modo de escrita e leitura. Esta configuração decorre de uma troca de tramas, a trama SET enviada pelo Emissor e a trama UA enviada pelo Recetor. Apesar de a função ser comum aos dois programas, há nela uma distinção das ações conforme o parâmetro status da struct referida no tópico anterior, recebida como parâmetro. - Leitura do ficheiro e armazenamento da sua informação num array de unsigned chars. - Criação do pacote de controlo Start, seguido do seu envio, já recorrendo à função llwrite(). - Criação dos pacotes de Informação, que resulta de uma divisão do array referido no segundo passo, e o seu respetivo envio, recorrendo também a llwrite(). - Criação do pacote de controlo End, seguido do seu envio, recorrendo à função llwrite(). - llclose(): Encerramento da ligação entre os dois computadores, através de uma troca de tramas. Neste caso, o Emissor receberá uma trama DISC, enviando como resposta outra trama DISC, e para terminar receberá uma trama UA.

Recetor: O executavel relativo ao Recetor exige tambem 2 argumentos: o nome da porta serie (por exemplo /dev/ttyS0), e o nome do ficheiro que vai ser transmitido (exemplo pinguim.gif) - llopen: Configura a ligação entre os dois computadores, abrindo a porta série em modo de escrita e leitura. Esta configuração decorre de uma troca de tramas, a trama SET enviada pelo Emissor e a trama UA enviada pelo Recetor. Apesar de a função ser comum aos dois programas, há nela uma distinção das ações conforme o parâmetro status da struct referida no tópico anterior, recebida como parâmetro. - Receção do pacote de controlo Start, recorrendo à função llread(). - Processamento do pacote de controlo Start recebido, de modo a receber corretamente o nome e o tamanho do ficheiro que vai ser copiado, para efeitos de apenas informar o utilizador destes dados. - Receção dos pacotes de Informação, recorrendo à função llread(). A cada pacote lido, a sua informação é processada (de modo a ficar apenas com os bytes de informação do ficheiro), e esta informação é logo de seguida escrita para o novo ficheiro, criado imediatamente antes desta receção. - Quando recebe o pacote de controlo End, o loop de receção de tramas termina. - llclose(): Encerramento da ligação entre os dois computadores, através de uma troca de tramas. Neste caso, o Recetor enviará uma trama DISC, recebendo como resposta outra trama DISC, e para terminar enviará uma trama UA.

4 Casos de uso principais

4. Casos de uso principais (identificação; sequências de chamada de funções)

5 Protocolo de ligação lógica

5. Protocolo de ligação lógica (identificação dos principais aspectos funcionais; descrição da estratégia de implementação destes aspectos com apresentação de extratos de código)

6 Protocolo de aplicação

6. Protocolo de aplicação (identificação dos principais aspectos funcionais; descrição da estratégia de implementação destes aspectos com apresentação de extractos de código)

7 Validação

7. Validação (descrição dos testes efectuados com apresentação quantificada dos resultados, se possível)

8 Eficiência do protocolo de ligação de dados

8. Eficiência do protocolo de ligação de dados (caraterização estatística da eficiência do protocolo, feita com recurso a medidas sobre o código desenvolvido. A caracterização teórica de um protocolo Stop Wait, que deverá ser usada como termo de comparação, encontra-se descrita nos slides de Ligação Lógica das aulas teóricas).

9 Conclusões

9. Conclusões (síntese da informação apresentada nas secções anteriores; reflexão sobre os objectivos de aprendizagem alcançados)

- Anexo I - Código fonte - Outros anexos, se necessário

b) O estudante tem a liberdade de "fundir"algumas das secções apresentadas (Sec. 2 a Sec.8). Poderá, por exemplo, despromover algumas das secções a subsecções.

c) O relatório não pode exceder 8 páginas A4, fonte de 11pt. Os anexos não estão incluidos nestas 8 páginas.

d) O relatório pode ser entregue até uma semana depois da demonstração ter sido aceite pelo professor.

e) Os relatórios devem ser submetidos via moodle, na turma correspondente. O grupo de alunos deverá, no entanto, facultar uma cópia do relatório em papel ao professor, caso este lhe seja pedido.

f) Algumas dicas sobre o estilo de escrita a adoptar em relatórios técnicos:

1. a escrita deve ser organizada em torno do parágrafo. Cada parágrafo deve ser dedicado a um único assunto e, na primeira linha do parágrafo, deve poder perceber-se de imediato qual é esse assunto;

2. o parágrafo deve ser grande. Este estilo de escrita permite uma leitura "fotográfica"da página;

3. escreva pela positiva; descreva o que fez e não o que não fez;

4. seja cuidadoso com a utilização de adjetivos e evite descrições vagas. Não diga que o seu código é rápido; diga que determinada função é executada em cerca de 200 ms. Não diga que fez muito código; diga que fez cerca de 800 linhas de código;

5. depois de escrever uma frase ou um parágrafo releia-o. Se conseguir colocar a mesma informação num menor número de palavras, faça-o;

6. Não use uma segunda frase para explicar de outra forma o assunto que descreveu na primeira frase; nesta situação deverá re-escrever a primeira frase;

7. Se, durante a escrita, sentir necessidade de abrir parenteses ou introduzir um parágrafo adicional para introduzir um qualquer conceito, reformule o texto. Provavelmente estes conceitos deveriam ter sido introduzidos numa secção ou num capítulo anterior. As frases e os parágrafos devem aparecer sempre em sequência natural;

8. Escreva o texto como se estivesse a fazer software. Defina nomes e conceitos antes de os usar e depois, no texto, use sempre o mesmo nome para o mesmo conceito, mesmo que tenha de o usar múltiplas vezes na mesma frase.