**Protocolo de ligação de dados**

**Relatório final**



Mestrado Integrado em Engenharia Informática e Computação

Redes de Computadores

**Turma 1 Grupo 4:**

Duarte Pinto Valente – up201504327

Nuno Miguel Outeiro Pereira – up201506265

Rui André Rebolo Fernandes Leixo – up201504818

Faculdade de Engenharia da Universidade do Porto

Rua Roberto Frias, sn, 4200-465 Porto, Portugal

3 de Novembro de 2017

Índice

Sumário \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ pág 3

Introdução \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ pág

Arquitetura \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ pág

Estrutura do código \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ pág

Casos de uso principais \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ pág

Protocolo de ligação lógica \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Protocolo de aplicação \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Validação \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Eficiência do protocolo da ligação de dados \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Conclusões \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Anexos \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Sumário

O âmbito deste relatório, com recurso ao trabalho elaborado, é suportar o projeto do protocolo de ligação de dados, que consiste em efetuar transferência de ficheiros entre dois computadores, servindo-se de uma ligação por cabo entre as portas de série de cada um dos computadores.

O trabalho realizado é capaz de transferir o ficheiro sem perdas, mesmo quando é introduzido lixo durante a transferência através de curtos-circuitos causados propositadamente, ou até interrupção do funcionamento da porta de série, sendo que no primeiro caso, o programa ignora o lixo introduzido e no segundo retoma a transferência quando a ligação for reestabelecida.

Introdução

Este trabalho tinha como principal objetivo a implementação de um protocolo de ligação de dados em linguagem C, em ambiente LINUX, onde era requerida a transmissão de ficheiros, sendo que esta era organizada por diferentes tipos de trama, tanto de informação como de supervisão.

O tipo de comunicação utilizado através da porta de série modelo RS-232 neste trabalho é em série assíncrona, sendo que foram implementadas várias metodologias de deteção e correção de erros, tais como a interrupção da transmissão por falha da porta de série, ou por ruído introduzido nas tramas através de curtos circuitos causados na ligação. O programa possui também um alarme caso o tempo de receção de uma trama exceda o máximo pré-estipulado, assim como uma condição de término do mesmo caso se exceda o número máximo de tentativas para enviar a mesma trama.

Este relatório vai ser subdividido em várias secções, de forma à informação ser mais facilmente acessível e percetível, sendo as seguintes:

**- Introdução** onde os principais objetivos do trabalho são apresentados

**- Arquitetura**

**- Estrutura do código** onde são apresentadas as funções de maior relevância, assim como estruturas de dados criadas de forma a facilitar a implementação de outras funcionaliades

**- Casos de uso principais**

**- Protocolo de ligação lógica**, descrição da estratégia de implementação da linkLayer

**- Protocolo de aplicação**, descrição da estratégia de implementação da appLayer

**- Validação**, onde serão alvo de atenção os testes realizados ao programa

**- Eficiência do protocolo de ligação de dados**, cálculos de eficiência através de estatísticas retiradas a partir de testes efetuados ao programa

**- Conclusão**

Arquitetura

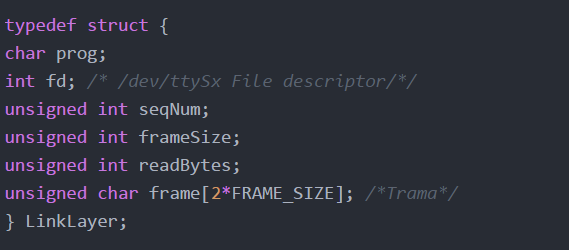
A maior divisão deste trabalho encontra-se precisamente na sua estrutura, dado que está organizado em duas camadas, a camada de ligação de dados (linkLayer) e a camada de aplicação (appLayer).

A camada de ligação de dados está associada à ligação à porta de série, incluindo a sua abertura e configuração de forma a preparar o seu correto funcionamento. É nesta camada também que é efetuada a implementação do envio e receção de mensagens e comandos. Processos de stuffing e destuffing são também implementados nesta camada.

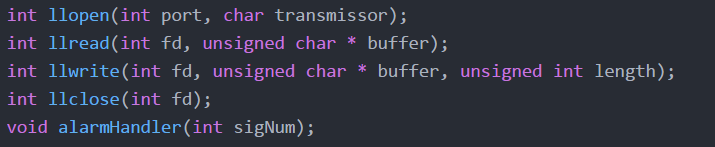
A camada de aplicação é dependente da camada de ligação, visto que usa e chama muitas das funções implementadas nessa camada. Esta camada tem uma grande influência no rumo que o programa toma, dado que é nesta que se executam as funções de emissão e receção de tramas.

Estrutura do código

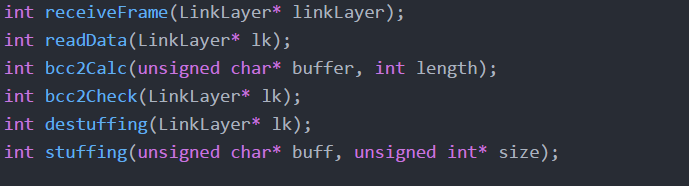
Na camada da ligação de dados está definida uma estrutura de dados chamada LinkLayer que contem o descritor de ficheiro, o tamanho da trama e a própria trama.



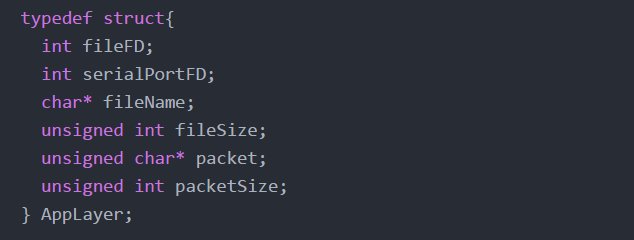
As principais funções implementadas nesta camada são: a função que estabelece a ligação, a função de leitura da trama, assim como a de escrita, a função que termina a ligação e a função que especifica o funcionamento do alarme.



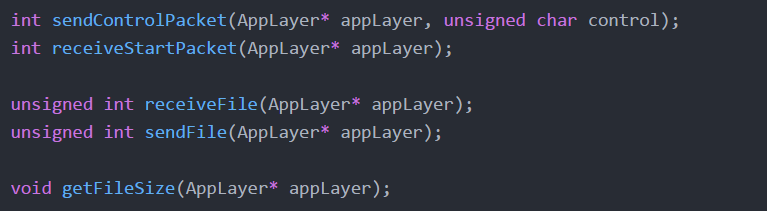
Outras funções também essenciais como as responsáveis pelo stuffing e destuffing, assim como pela receção da trama são também nesta camada implementadas.



No ficheiro da appLayer.h é possível encontrar a estrutura de dados que representa a camada de aplicação, sendo que esta é composta por um descritor de ficheiro da porta de série, descritor, nome e tamanho do ficheiro a transmitir, assim como o tamanho do pacote a transmitir de cada vez.



As funções implementadas nesta camada são as seguintes:



Protocolo de ligação lógica

Esta é a camada responsável por estabelecer a ligação através da porta de série, efetuando-se a partir da função llopen.

Esta função envia um comando SET e aguarda um comando UA como resposta do recetor, que serve para confirmar que recebeu um comando SET. Este comando UA tem de ser recebido pelo emissor dentro de um tempo limite estipulado pelo utilizador no início do programa, pois, caso contrário, um alarme será ativado. Após o aviso do alarme ter ocorrido, o comando SET será reenviado, repetindo todo o procedimento anterior. Este reenvio do comando SET tem também um número máximo de tentativas estipulado no programa e caso o número de tentativas esgote o programa irá terminar com erro. Caso o comando UA seja recebido com sucesso pelo emissor significa que a ligação foi estabelecida e que o programa deverá continuar a sua execução.

De seguida, a função llwrite é chamada para enviar a trama ao recetor e fica a aguardar uma resposta por parte deste para poder optar pela próxima ação. Se não for obtida uma resposta durante o período definido, o que se segue é semelhante ao caso anterior do estabelecimento da ligação sendo que neste caso a trama será reenviada até se ter atingido o numero máximo de tentativas, sendo que neste ponto, o programa termina com retorno de erro. Caso não se chegue a atingir este caso e de facto o emissor receba uma resposta, então uma ação será tomada conforme a resposta. Caso esta seja uma trama RR, então tudo correu conforme planeado e a trama foi recebida com sucesso, caso a resposta seja uma trama REJ, a trama deve ser reenviada.

Outra função importante para este processo é a llread que fica num ciclo a aguardar a receção da trama. Esta função vai chamar uma outra função denominada receiveFrame, onde se encontra a máquina de estados para analisar a trama que está a ser recebida, sendo que na eventualidade de serem encontrados bytes não esperados na trama, voltar ao início da máquina de estados e tentar encontrar o pedaço de informação certo. Existe também uma outra função chamada neste processo, readData, que vai ler o pedaço da trama que contém a informação do ficheiro, chamando também a função responsável por fazer o destuffing da informação. A seguir, seguem-se as verificações do BCC2 e se tudo corroborar a função envia como resposta um RR com o número da sequência seguinte e guarda a informação que recebeu. Caso este processo de verificação falhe, é enviado um REJ com o número da sequência da trama que acabou de ser rejeitada, para ser reenviada. Há ainda a possibilidade de a trama recebida conter o comando DISC, sinal de que a ligação deve ser fechada.

O comando DISC, referido aqui anteriormente, é gerado pela função llclose que tem como objetivo terminar a ligação, enviando para tal efeito, o comando DISC ao recetor, sendo que para a ligação ser terminada com sucesso, é também necessário que o recetor envie um UA e este seja recebido pelo emissor pois, de outra forma, tal como nas funções anteriores um alarme vai controlar o tempo decorrido e o número de tentativas para poder terminar com sucesso.

Protocolo de aplicação

Anexos