****

Mestrado Integrado em Engenharia Informática e Computação

*Redes de Computadores*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Protocolo de Ligação de Dados***

Trabalho 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Segunda-feira, 10 de Novembro de 2014

# Página de Rosto

Título do Projeto:

*Protocolo de Ligação de Dados*

Curso:

*Mestrado Integrado em Engenharia Informática e Computação*



Unidade Curricular / Ano Letivo:

*Redes de Computadores - 2014/2015*

Professor:

*Manuel Alberto Pereira Ricardo*

*Página pessoal:* [*http://paginas.fe.up.pt/~mricardo/*](http://paginas.fe.up.pt/~mricardo/)

*E-mail:* [*mricardo@fe.up.pt*](mailto:mricardo@fe.up.pt)



Elementos de Grupo:

Henrique Manuel Martins Ferrolho

[201202772](https://sigarra.up.pt/feup/pt/fest_geral.cursos_list?pv_num_unico=201202772)

[ei12079@fe.up.pt](mailto:ei12079@fe.up.pt)



João Filipe Figueiredo Pereira

[201104203](https://sigarra.up.pt/feup/pt/fest_geral.cursos_list?pv_num_unico=201104203)

[ei12023@fe.up.pt](mailto:ei12023@fe.up.pt)



José Pedro Vieira de Carvalho Pinto

[201203811](https://sigarra.up.pt/feup/pt/fest_geral.cursos_list?pv_num_unico=201203811)

[ei12164@fe.up.pt](mailto:ei12164@fe.up.pt)



Miguel Ângelo Jesus Vidal Ribeiro

[201100657](https://sigarra.up.pt/feup/pt/fest_geral.cursos_list?pv_num_unico=201100657)

[ei11144@fe.up.pt](mailto:ei11144@fe.up.pt)

Data de Entrega:

*Segunda-feira, 10 de Novembro de 2014*

Nota Final:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

# Sumário

No âmbito da unidade curricular de *Redes de Computadores*, do curso *Mestrado Integrado em Engenharia Informática e Computação*, foi-nos proposto a elaboração de uma aplicação cujos conteúdos, até à data, estavam a ser abordados nas aulas teóricas. Entre os conteúdos abordados destacam-se a Camada de Ligação de Dados (*Data Link Layer*) e a Camada Física (*Physical Layer*). A proposta de trabalho baseou-se em realizar uma transferência de ficheiros entre dois computadores. Utilizou-se para o caso, conceitos interiorizados nas aulas teóricas e laboratoriais acerca de ligação de dados, através de uma ligação por cabo pelas portas de série de cada máquina.

A elaboração de um relatório final para o trabalho tem como objetivo a consolidação do trabalho realizado ao longo desta primeira metade de semestre. Neste contexto pode-se afirmar que é necessário que haja uma ligação entre a parte prática e teórica que envolveu o projeto. Esta ligação permitirá ao docente uma avaliação correta e concisa, podendo-se dar destaque ao bom entendimento por parte do grupo das matérias lecionadas e da sua implementação.

# Introdução

O trabalho realizado ao longo das aulas laboratoriais da unidade curricular de Redes de Computadores tem como objetivo a implementação um protocolo de ligação de dados, de acordo com a especificação descrita no guião. Na sua especificação era pedida a combinação de características de protocolos de ligação de dados existentes, entre os quais, a transparência na transmissão de dados, uma transmissão organizada em diferentes tipos de tramas: tais como informação, supervisão e não numeradas, que eram protegidas por código detetor de erros. De referir que o tipo de transmissão utilizado em todo o projeto é em série assíncrona. Este modelo de transmissão foi aplicado nas primeiras aulas, onde nos foi pedido a sua implementação que serviria de esboço para o desenvolvimento de uma aplicação com outro nível de complexidade.

Outro objetivo do trabalho consistia em testar o protocolo com uma aplicação simples de transferência de ficheiros, igualmente especificada. Na especificação da aplicação de teste era pedido o suporte de dois tipos de pacotes enviados pelo emissor: o de dados e o de controlo.

Nos pacotes de controlo destacam-se o de *start*, sinalizando o início da transmissão;e o de *end*, sinalizando o fim da transmissão.

Já os pacotes de dados iriam conter frações do ficheiro a transmitir.

O relatório final deverá servir, assim como já foi dito em cima, na secção de Sumário para uma consolidação de matérias entre a parte teórica e prática do projeto. No relatório devem ser especificados todos os pontos descritos na implementação do trabalho e deve ser dividido em várias secções para uma boa organização e descrição das mesmas.

Assim sendo, o relatório subdivide-se em:

- **Introdução**, onde são indicados os objetivos do trabalho e relatório;

- **Arquitetura**, contendo esta secção a explicação dos blocos funcionais e interfaces;

- **Estrutura do Código**, fazendo-se referência às API’s consultadas e implementadas, a estruturas de dados elaboradas, às funções de maior importância e à sua relação com a Arquitetura;

- **Casos de uso Principais**, fazer a sua identificação e abordar as sequências de chamada de funções;

- **Protocolo de ligação lógica**, onde se descreve a estratégia de implementação abordada, com referência a excertos de código e identificação dos principais aspetos funcionais;

- **Protocolo de Aplicação**, onde a exemplo do protocolo de ligação lógica também poderá ser possível encontrar excertos de código, a sua explicação e a estratégia abordada na elaboração deste ponto do trabalho;

- Na **Validação** serão descritos os testes elaborados, ilustrando os seus resultados e possíveis comentários aos mesmos;

- Em **Elementos de valorização**, estarão descritos os pontos adicionais que eram possíveis de ser realizados no trabalho e que o grupo teve a competência de implementar;

- Em **Conclusões** estará presente o comentário e análise final do grupo perante este projeto.

Antes de prosseguirmos é de referir que o grupo desenvolveu o projeto pedido em ambiente LINUX, com linguagem de programação C e utilizando portas de série do modelo RS-232, cuja comunicação é assíncrona.

# Arquitetura

Nesta secção será descrita a arquitetura do trabalho e como ela foi planeada e desenvolvida. Haverá também uma abordagem às interfaces implementadas e respetiva explicação funcional.

O trabalho está organizado em dois blocos maioritários que permitem a correta funcionalidade pedida no guião. Os blocos com maior relevo são o *protocolo de ligação de dados* e o da *aplicação*, que estão implementados em ficheiros *source* e *header* (\*.c e \*.h). Os ficheiros *DataLink.c* e *Datalink.h* representam o bloco de ligação de dados enquanto os ficheiros *Application.c* e *Application.h* representam o bloco da aplicação.

Seguidamente iremos descrever cada um dos blocos e o seu funcionamento baseando-nos no que eles representam e executam na aplicação elaborada.

O bloco de ligação de dados disponibiliza funções genéricas do protocolo estando nele especificadas as funcionalidades de sincronismo e numeração de tramas, suporte de ligação, controlo da transferência e de erros, controlo de fluxo e todos os aspetos relacionados com a ligação à porta série tais como a sua abertura e definição das suas propriedades.

O bloco da aplicação não é independente do bloco de ligação de dados já que tira proveito das suas propriedades para a implementação das suas funções. Podemos dizer que o bloco da aplicação é o responsável pelo envio da transferência dos ficheiros pois é nele que são executadas as funções de receção e emissão de tramas. Os pacotes de controlo e de dados são processados e enviados neste bloco tornando-o uma peça fulcral no comando da aplicação em si.

No que diz respeito à *interface* da aplicação, ela encontra-se implementada nos ficheiros *CLI.c* e *CLI.h*. Nesta camada é tratada a interação com o utilizador da aplicação. Encontram-se elaboradas funções que permitem uma transferência instantânea com valores predefinidos pelo grupo e que respeitam o que era pedido no guião do trabalho. Se o utilizador colocar todos os parâmetros inicialmente, a aplicação irá tratar as questões de *baudrate*, tamanho máximo da mensagem, número de tentativas em caso de falha de comunicação e intervalo entre cada tentativa como foram especificados pelo grupo. Caso o utilizador não coloque parâmetros e apenas execute normalmente a aplicação então é requisitada ao utilizador a inserção manual dos valores lá descritos. Para ajuda ao utilizador, o grupo fornece na mensagem lá representada os valores válidos para inserção nos parâmetros que assim o exijam. Após a inserção manual dos valores a aplicação é executada com as propriedades definidas pelo utilizador e a transferência é executada.

É de referir que o bloco da *interface* interage com o bloco da aplicação com o objetivo de iniciar o programa com valores editados pelo utilizador. Esta interação basicamente inicia o construtor da aplicação com outros valores que não os predefinidos.

# Estrutura do código

# Casos de uso principais

# Protocolo de ligação lógica

# Protocolo de aplicação

A aplicação é a camada de mais alto nivel, responsável pelo envio e receção do ficheiro escolhido. Esta camada vai atuar tanto ao nivel do recetor como do transmissor e é responsável por interpretar os inputs do user da seguinte forma:

**Transmissor:**

Iniciamos a ligaçao chmando a funçao *llopen,* de seguida éaberto o ficheiro que pertendemos enviar e calculado o seu tamanho.

Depois disto é chamada a função *sendControlPackage* com a variavel *CTRL\_PKG\_START,* ( pacote de controlo Strat), tamanho e nome do ficheiro a enviar.

Após enviar o pacote de controlo START, é chamada a função *sendDataPackage*, função esta responsavel pelo envio do ficheiro.

Tendo terminado com sucesso o envio dos dados, é fechado o ficheiro e é chamada outra vez a função *sendControlPackage* desta vez com a variavel CTRL\_PKG\_END que indica um pacote de controlo END e finalizamos chamando a função *llclose*.

**Recetor:**

Inicialmente establecemos a ligação chamando a função llopen de seguida é chamada a função *receiveControlPackage* que recebe e verifica o pacote de controlo Start. Após isso é chamada a funçao *receiveDataPackage* que vai tratar de toda a receçao e verificação do ficheiro enviado.

Por fim é chamada a funçao *receiveControlPackage* que recebe o pacote de controlo End e termina o chamando a função *llclose*.

***sendControlPackage(int fd, int C, char\* fileSize, char\* fileName)* -** Usada pelo transmissor para enviar um pacote de controlo (START se “C” for a variavel CTRL\_PKG\_START, ou END se “C” for a variavel CTRL\_PKG\_END) com algumas informações acerca do ficheiro. Usa a função *llwrite* do protocolo de ligação para envio das tramas.

***sendDataPackage(int fd, int N, const char\* buf, int length) -*** Usada pelo transmissor para enviar o ficheiro para o outro computador. Lê do ficheiro um determinado número de bytes e coloca-os no campo de dados do pacote, de seguida chama a função *llwrite* para escrever esse pacote. Este processo é repetido até o ficheiro ser todo transferido.

***receiveDataPackage(int fd, int\* N, char\*\* buf, int\* length) -*** Usada pelo Recetor para tartar da receção do conteudo do ficheiro. Usa a função *llread* do protocolo de ligação para ler o pacote.

***receiveControlPackage(int fd, int\* ctrl, int\* fileLength, char\*\* fileName) –*** Usada pelo recetor para receber e ler os pacotes de controlo Start e END.

**CTRL\_PKG\_START =** *1*

***CTRL\_PKG\_END =*** *2*

# Validação

# Elementos de valorização

# Conclusões

Após a realização do trabalho, sentimos que os objetivos inicialmente propostos foram cumpridos e a independência entre as camadas de aplicação e ligação foi assegurada. Todos os aspetos pedidos, inclusive, alguns dos pontos de valorização, foram implementados estando a funcionar corretamente.

Tentamos ao máximo em todas as situações cumprir o protocolo proposto no guião, implementando também algumas estruturas próprias que facilitaram a implementação do nosso código.

Com a realização deste trabalho, ficamos assim mais habitados em redes de computadores.

Do ponto de vista ampliação do conhecimento pessoal, ficamos muito mais habilitados em redes de computadores, pois, com a experiência adquirida durante o desenvolvimento deste trabalho, utilizámos mecanismos que se assemelham em muitos aspetos a mecanismos de comunicação de computadores, que utilizamos no quotidiano.

Além disso, a resolução de erros, que por vezes tivemos de efetuar, fez-nos adquirir alguns conhecimentos muito mais pormenorizados e que, antes de fazermos o trabalho, não tínhamos.