Universidade de Brasília (UnB),
Campus Darcy Ribeiro

Simulador de Mensagens Utilizando as Camadas Física e de Enlace

Teleinformática e Redes 1 - T01
Prof. Marcelo Antônio Marotta

Membros
Arthur de Sá Antero - *212006577*
Paulo Fernando Vilarim de Almeida - *211043351*
Jéssica Leal de Melo - *211038253*

1. Introdução

O projeto desenvolvido utiliza-se da troca de mensagens em um chat, hospedado em um servidor em que se pode ter qualquer quantidade de usuários participando simultaneamente, com o objetivo de simular na camada de aplicação as etapas de comunicação que perpassam pelas camadas física e de enlace. A pessoa que envia uma mensagem neste grupo pode escolher mandar uma mensagem que passa pela simulação das etapas das camadas física e de enlace da teoria de redes, ou simplesmente mandar uma mensagem de forma direta utilizando socket.

Ao iniciar um código de interface, primeiro é pedido para se identificar com um nome, para que os outros usuários possam identificar quem está mandando a mensagem.

Para enviar mensagens, tem-se 2 entradas de texto para envio de mensagens, em que a primeira é utilizada para a mensagem em si, e a segunda serve para digitar ou não comandos acerca das etapas das camadas física e de enlace.

2. Implementação

Para rodar o projeto, foi realizado um shell script, denominado “script.sh”, o qual utiliza instâncias de terminal (com o gnome-terminal) para rodar os arquivos necessários do projeto simultaneamente, sendo estes o “simulador.py” (servidor) e dois agentes para se comunicar, ou seja, duas instâncias de “interfaceGUI.py” (transmissor/receptor).

Um diagrama de funcionamento do projeto segue abaixo:

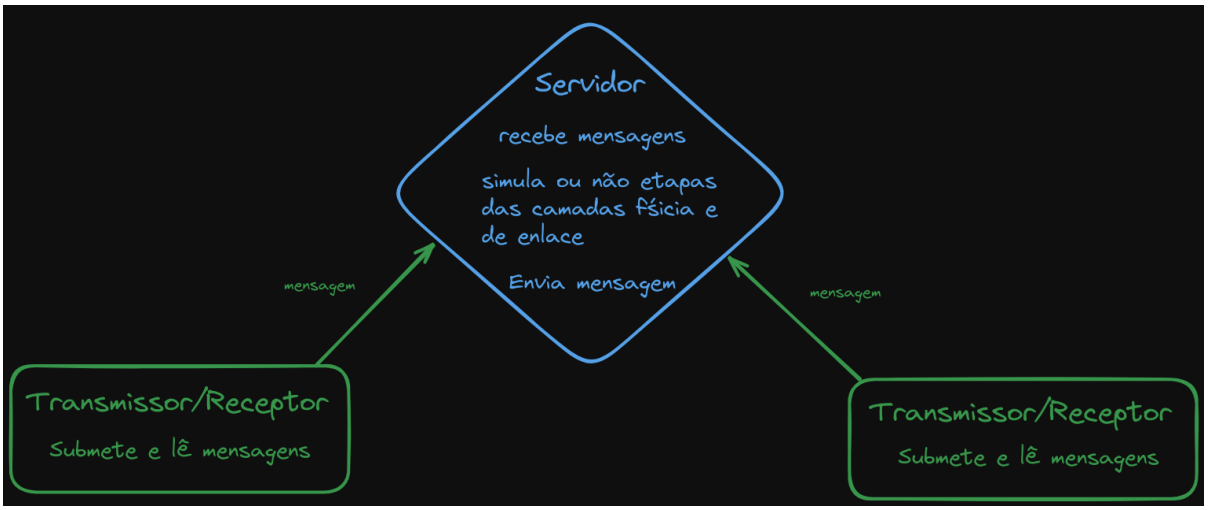

Figura 1. Diagrama de funcionamento e interação das partes que compõem o projeto, no objetivo de simular as etapas e os protocolos da camada física e de enlace da teoria de redes .

Caso o usuário não possua “gnome-terminal”, é recomendado rodar manualmente os arquivos, tendo em mente que o “simulador.py” deve estar operando para a devida inicialização do “interfaceGUI.py”.

Ao entrar no chat com um nome de usuário escolhido pela pessoa, o servidor/simulador envia ao usuário um passo a passo de como utilizar os comandos. O primeiro campo de entrada de texto (de cima para baixo) é utilizado unicamente para o envio da mensagem que se deseja enviar, enquanto o segundo campo de entrada de texto serve para utilização de comandos, os quais servirão para estabelecer quais protocolos da camada física

e de enlace serão simulados no envio da mensagem; assim, o campo de comandos deve ser utilizado para passar, respectivamente, a modulação digital pedida, a modulação de portadora, o método de enquadramento de dados e o método de detecção de erros.

Assim que um usuário logar no servidor e escolher seu nome de usuário, a página de interface inteira que aparecerá a ele será a mostrada na seguinte imagem:

Figura 2. Interface de usuário após logar como nome de usuário, mostrando os dois terminais e mensagens iniciais.

Os comandos não podem fugir de como são mostrados na imagem acima, caso fujam resultará em erro no envio da mensagem. Caso um usuário queira mandar uma mensagem que não passe por todos esses protocolos, basta mandar uma mensagem no primeiro campo de entrada de texto, com o segundo vazio.

Outrossim, ao se escolher os protocolos que você quer utilizar para enviar a mensagem, o servidor irá simular a seguinte ordem de protocolos a serem feitos antes da mensagem ser realmente enviada ao chat via socket:

- Transmissor: Enquadramento -> Detecção de erros -> Correção de erros -> Modulação digital -> Modulação de portadora
- Receptor: Correção de erros -> Detecção de erros -> Desenquadramento

Escolhidos os protocolos, a mensagem a ser enviada será convertida em uma string de bits (seguindo a tabela ASCII) e será manipulada de acordo com os algoritmos de cada etapa, além de serem plotados gráficos para exibição da modulação realizada. Após a execução dos algoritmos no lado transmissor, a mensagem é recuperada com a execução dos algoritmos escolhidos no lado receptor, para então a mensagem ser enviada ao grupo, da mesma maneira que foi submetida, sem que haja perda de informação.

Para o usuário que enviar a mensagem, a interface irá mostrar como a mensagem está ficando em bits após cada um dos passos da parte da transmissão primeiro, transformando a mensagem em bits, depois enquadrando, logo após fazendo a detecção de erro e por último passando pelo método de Hamming, o que resultará em algo dessa forma:

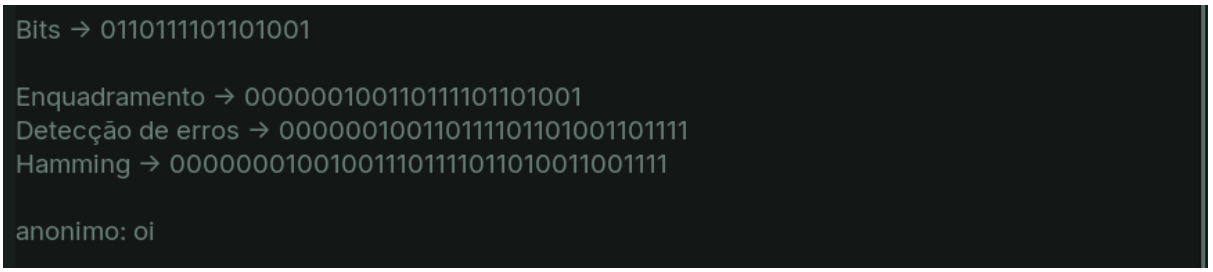

Figura 3. Exemplo de envio da mensagem "oi" com escolha de enquadramento de contagem de caracteres e detecção de erros CRC.

Ainda na parte de transmissor, tem-se as modulações digital e por portadora, respectivamente, que mostram os seguintes gráficos do mesmo exemplo de mensagem anterior:

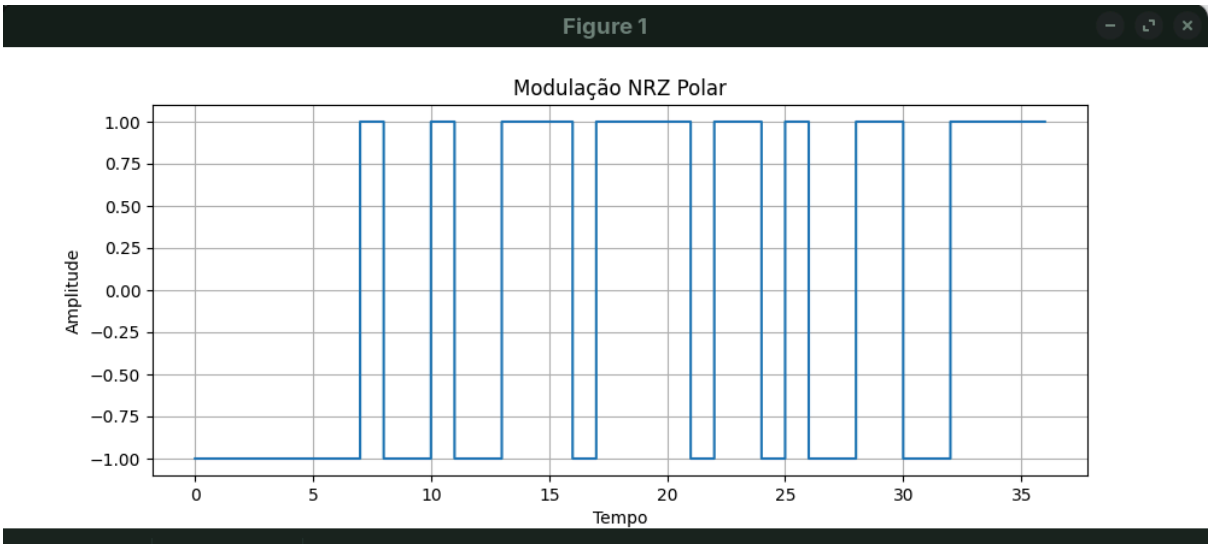

Figura 4. Modulação digital pelo método de NRZ Polar após enquadramento e gestão de erros da mensagem "oi".

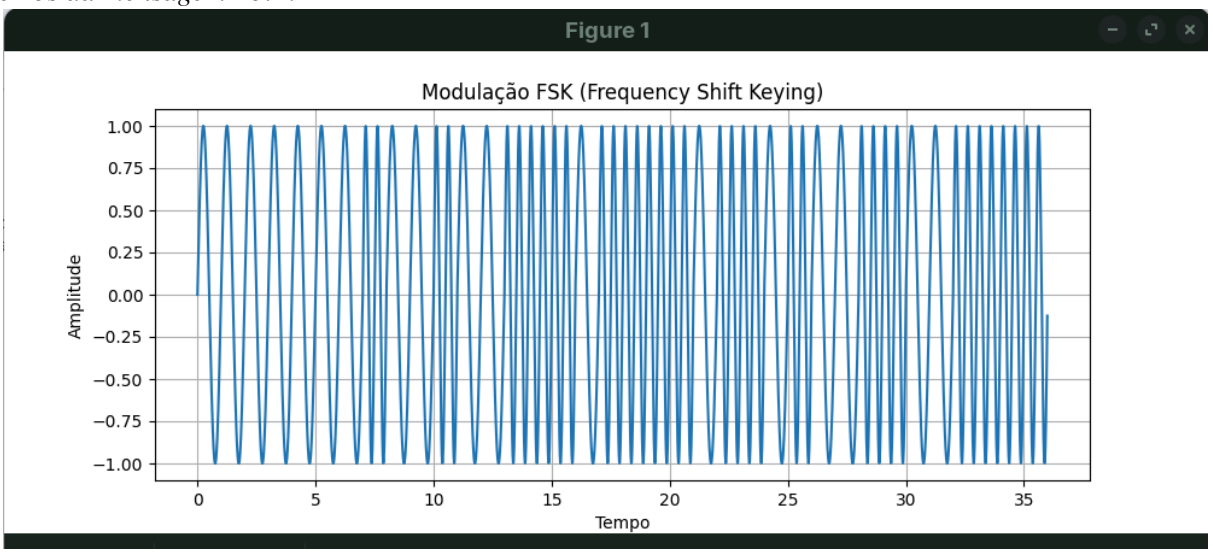

Figura 5. Modulação de portadora pelo método FSK após enquadramento e gestão de erros da mensagem "oi".

No que tange o funcionamento dos protocolos pedidos pelo projeto, temos, primeiro, na parte física, as partes das modulações digitais e de portadora, assim o funcionamento de cada uma se resume a:

- Modulação digital NZR-Polar: Consiste na criação de um gráfico digital em que os 1's que entram na modulação são passados agora como amplitude V (escolhida na hora de criar a modulação) e os 0's são passados no gráfico com a amplitude de -V.

- Modulação digital Manchester: Consiste na criação de um gráfico digital em que faz-se uma operação digital XOR da entrada de bits com um clock pré-definido, o gráfico gerado será o resultado dessa operação digital. O clock observado nessa modulação deve ter metade da frequência do sinal, para que a entrada seja dividida em duas partes na hora de colocar no gráfico.

- Modulação digital Bipolar: Consiste na criação de um gráfico digital em que os 1's de entrada ficam alternando entre amplitude $\pm$V a depender do valor anterior da entrada 1, e quando a entrada consiste em um 0, o valor de saída do gráfico digital é apenas ele mesmo.

- Modulação de portadora ASK (Amplitude Shift Keying): Consiste na criação de um gráfico em que os 1 's da entrada representam uma senoidal na modulação e os 0 's representam a ausência de amplitude no gráfico.

- Modulação de portadora FSK (Frequency Shift Keying): Consiste na criação de um gráfico em que os 1 's da entrada representam uma senoidal com uma frequência escolhida na saída, e os 0 's geram a mesma senóide, mas com uma frequência diferente da escolhida para o gráfico gerado.

- Modulação de portadora 8-QAM: Consiste em uma técnica de modulação em que são gerados duas ondas portadoras em fase, cossenoidal, e em quadratura, senoidal, em que o resultado é uma soma dessas duas ondas. Para a geração dessas duas ondas

faz-se uma constelação obedecendo a codificação de Grey e mudando apenas 1 bit por vizinho, pegando sempre de 3 em 3 bits para gerar esse gráfico, pois $2^3 = 8$.

Quanto à camada de enlace, temos as etapas de enquadramento, detecção de erros e possível correção de erros.

A começar pela análise do enquadramento, esta etapa resume-se em pegar uma sequência de bits, proveniente da camada adjacente, e particioná-los em quadros. Já no desenquadramento, ocorre o recebimento destes quadros e a então recomposição dos bits originais. Neste tema, podemos elucidar o funcionamento dos seguintes protocolos:

- Contagem de Caracteres: este protocolo fundamenta-se em enquadrar bits com a adição de cabeçalhos os quais representam a quantidade de bits ou bytes que o quadro possui, delimitando assim o início e o fim de um quadro. No projeto desenvolvido, foi optada a utilização do algoritmo de Contagem de Bytes;

- Inserção de Caracteres: este protocolo fundamenta-se em enquadrar bits com a inserção de caracteres especiais que delimitam o início e o fim de um quadro (FLAG), e, para caso o caractere especial venha de maneira indesejada no meio de um quadro, é adicionado um outro caractere especial logo antes (ESC), para manter a devida leitura do quadro. No projeto desenvolvido, foi escolhida a utilização do algoritmo de Inserção de Bytes com os bytes de FLAG e ESC sendo os caracteres “&” (00100110) e “ESC” (00011011), respectivamente.

Outrora, a Detecção de Erros na camada de enlace resume-se em utilizar bits de validação nos quadros, para verificar se ocorreu um erro durante a transmissão dos bits dos quadros. Neste tema, podemos elucidar o funcionamento dos seguintes protocolos:

- Bit de Paridade: este protocolo fundamenta-se na inserção de um único bit ao final. Este bit irá complementar a paridade dos demais bits que o antecedem. Por exemplo, para o algoritmo de Bit de Paridade Par, o bit ao final será acrescido de forma que a quantidade de 1 's seja par, enquanto que para o algoritmo de Bit de Paridade Ímpar, o bit ao final será acrescido de forma que a quantidade de 1 ‘s seja ímpar. Dessa forma, ao chegarem todos os bits ao receptor, este, por sua vez, irá verificar se a paridade segue como desejado, de forma que tanto transmissor quanto receptor estejam seguindo o mesmo algoritmo de paridade. No projeto desenvolvido, foi implementado o algoritmo de Bit de Paridade Par;

- CRC: o protocolo CRC fundamenta-se na divisão dos bits que serão validados (acrescidos de zeros à direita, na mesma quantidade do divisor menos 1) por um outro determinado conjunto de bits (representativos de um polinômio gerador, desde que o primeiro e o último bits sejam 1). Após a divisão no lado transmissor, o resto da divisão será incorporado ao dividendo e este será o novo conjunto de bits a ser transmitido. Já no lado receptor, este, por sua vez, irá realizar a divisão dos bits que chegaram pelo polinômio gerador e, caso o resto da divisão agora não seja zero, já que o resto foi incorporado antes da transmissão, então é identificado que ocorreu um erro na transmissão dos bits. No projeto desenvolvido, foi implementado o algoritmo de CRC com o polinômio gerador sendo representado pelos bits “1000111”.

Por fim, a Detecção e Correção de Erros, comumente usada em canais com pouco ruído, resume-se em utilizar bits de validação nos quadros, para verificar se ocorreu um erro durante a transmissão dos bits, bem como também aplicar um algoritmo para identificação específica do erro, a fim de se aplicar uma correção. Neste tema, podemos elucidar o funcionamento do seguinte protocolo:

- Código de Hamming: o algoritmo de Hamming fundamenta-se na inserção de bits de verificação em posições pré-determinadas (potências de 2) ao conjunto de bits originais. Após a inserção, cada bit do conjunto original terá uma nova posição, e esta posição será fatorada também em potências de 2, de forma que este bit de dados seja constituinte do cálculo do bit de verificação o qual apareceu em sua fatoração. Dessa maneira, cada bit de verificação terá um conjunto de bits originais de dados para participar de seu cálculo, e esse cálculo segue um algoritmo de paridade. Por exemplo, para o cálculo do bit de posição 4, serão vistos os bits constituintes de seu cálculo e, caso a paridade do algoritmo pré-acordada seja par, o bit desta posição 4 terá que assumir um valor de forma que a quantidade de 1 ‘s, no conjunto do bit de verificação com os bits constituintes do cálculo, seja par. No lado do receptor, ele irá recalcular as paridades dos bits de verificação com os bits constituintes de seus cálculos e, caso alguma paridade não esteja correta, será detectado um erro. Ademais, será possível identificar o erro ao se ler os bits de verificação da maior para a menor posição. No projeto desenvolvido, o algoritmo de Hamming implementado foi o Código de Hamming de paridade Par.

3. Membros

Arthur de Sá Antero: Modulação de portadora 8-QAM, criação da interface utilizando GTK, criação da funcionalidade do socket do arquivo de simulador.py, integração de todas as camadas com o código do simulador, documentação, padronização de retorno de funções, script bash para linux com terminal gnome, relatório do projeto.

Paulo Fernando Vilarim de Almeida: Código de detecção e correção de erros de Hamming, código de detecção de erros CRC, padronização de retorno de funções, parte receptora dos códigos de enquadramento Contagem de Bytes e Inserção de Bytes, refatoração final de código, relatório do projeto.

Jéssica Leal de Melo: Todos os códigos das três modulações digitais, códigos das modulações de portadora ASK e FSK, parte transmissora dos enquadramentos Contagem de Bytes e Inserção de Bytes, código de detecção de erro Bit de Paridade par.

4. Conclusão

No fim, pode-se afirmar que o simulador pedido para o trabalho da disciplina cumpriu o seu papel, fazendo que a mensagem passe pelas etapas das camadas física e de enlace de forma tranquila e mostre, por meio de gráficos e mensagens, como cada etapa funciona.

Algumas das dificuldades enfrentadas no desenvolvimento do projeto foram:

- Integração dos códigos de cada membro do grupo por uma falta de padrão de retorno das funções.
- Entendimento do processo de etapas em que a mensagem iria passar.
- Falta de entendimento da biblioteca de interface GTK.
- Falta de entendimento da biblioteca do Matplotlib, resultando em frequências altas nas modulações.