|  |  |
| --- | --- |
| D:\imagens\design\logos\uern.png | Governo do Estado do Rio Grande do Norte  Secretaria de Estado da Educação, da Cultural e dos Desportos – SECD  UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE – UERN  Pró-Reitoria de Ensino e Graduação – PROEG  Ciências da Computação  Trabalho prático para a 2ª e 3ª unidade |

**TRABALHO PRÁTICO DE IMPLEMENTAÇÃO**

**1. Informações gerais**

O objetivo deste trabalho é o de se desenvolver um sistema que possa gerenciar a movimentação de trens dentro de uma rede ferroviária simulada. A rede ferroviária em questão é composta por seis malhas tal como mostrado na Figura 1.

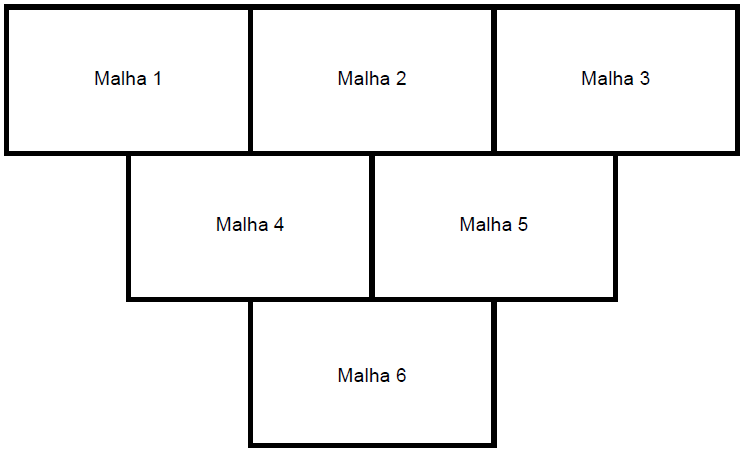


Figura 1 Rede ferroviária a ser simulada

Todas as malhas são retangulares, com dimensões de 10 x 6 UC (UC = unidades de comprimento = metro, quilometro ... ). Cada lado do retângulo da malha é formado por um ou dois ramos, sendo o ramo um trecho de linha ferroviária entre dois entroncamentos (junção de ramos). Cada trem sempre circula na mesma malha e não é possível colocar mais de um trem por malha, de tal modo que o número máximo de trens que podem estar trafegando na rede simultaneamente é de 6 (seis). Não é permitida a presença de mais de um trem em um mesmo ramo, mesmo que estejam espaçados de tal forma a evitar colisões.

As características dos trens devem ser definidas no instante em que eles comecem a trafegar na rede, não variando posteriormente. Tais características podem ser:

* Comprimento: cada trem pode ser composto por 1, 2 ou 3 vagões em que cada vagão tem comprimento de 1 UC
* Velocidade: o trem pode trafegar com qualquer velocidade entre 0,5 e 2,0 UC/s, inclusive
* Sentido: os trens podem se mover no sentido horário ou anti-horário

O trabalho poderá ser desenvolvido de maneira individual ou em grupo de até duas pessoas, sendo entregue **até as 23:59h da data estabelecida** em sala de aula. **Não haverá prorrogação deste prazo** e os trabalhos entregues a partir das 00:00h do dia seguinte àquele estabelecido, terão sua pontuação reduzida conforme equação abaixo:

em que se refere aos dias de atraso, contados a partir das 00:00h do dia subsequente aquele estabelecido e corresponde ao -ésimo critério acima estabelecido.

Todos os trabalhos deverão ser enviados por e-mail para diogolr@gmail.com com o assunto [CCNC] Trabalho Trens – Grupo X, para os alunos da turma do Núcleo de Nova Cruz e com o assunto, [CCSC] Trabalho Trens – Grupo X, para os alunos da turma do Núcleo de Santa Cruz.

**2. Características do sistema a ser desenvolvido**

O sistema a ser desenvolvido deverá ser composto por duas unidades de *software*, denominadas **gerenciador** e **lançador**, cujas características são descritas a seguir.

**2.1 Gerenciador**

O gerenciador será responsável pelas seguintes funções:

* Criar e destruir todas as estruturas de controle necessárias para a comunicação inter-processos (semáforos, memórias compartilhadas ...)
* Permitir a visualização do estado do sistema através de uma interface gráfica
* Checar a consistência do sistema, detectando situações catastróficas (colisões, trens no mesmo ramo ...)
* Calcular a velocidade média dos trens

As estruturas de controle serão criadas no instante do lançamento do gerenciador e destruídas quando o programa for encerrado. Dentre as estruturas necessárias, estão incluídas a(s) memória(s) compartilhada(s), que conterão as informações sobre os trens, os semáforos, necessários para gerenciar o acesso aos dados e as colisões entre os trens, e o servidor TCP (3ª unidade), para gerenciar a comunicação via rede.

O gerenciador funcionará com ciclos de 250 ms. A cada ciclo o *software* irá ler as informações da(s) memória(s) compartilhada(s) sobre a existência ou não dos trens em cada uma das 6 malhas e, em havendo, os dados sobre sua localização.

Com base nas posições dos trens, o gerenciador deverá exibir o estado e localização dos trens na interface gráfica, atualizando e exibindo a velocidade média de cada um deles, e verificar a existência de situações catastróficas, tais como: colisão entre trens ou a presença de mais de um trem por ramo. Sendo detectada alguma destas situações, o gerenciador deverá exibir uma mensagem de alerta e interromper o funcionamento do sistema.

O gerenciador permanecerá funcionando até que o usuário deseje encerrar a aplicação.

**2.2 Lançador**

O lançador fornecerá ao usuário uma interface (textual ou gráfica) para introduzir ou retirar trens do sistema. Na remoção do trem, o único dado necessário é o número da malha. Por outro lado, ao adicionar o trem o usuário deverá que fornecer as seguintes informações:

* Número da malha
* Número de vagões (1, 2 ou 3)
* Velocidade (0,5 a 2,0 UC/s)
* Sentido (horário ou anti-horário)
* Ramo de entrada: esquerdo, superior esquerdo, superior direito, direito, inferior direito ou inferior esquerdo. O trem sempre inicia no centro do ramo escolhido

O lançador só poderá ser iniciado quando o gerenciar estiver em funcionamento e permanecerá funcionando até que o usuário deseje encerrá-lo ou até que o gerenciador seja encerrado.

Cada trem lançado na rede deverá corresponder a um “processo” separado, criado ou destruído quando o usuário solicita ao lançador a inclusão ou remoção do trem, respectivamente. Ao encerrar o lançador, os “processos” que representam os trens em funcionamento deverão ser encerrados.

Os “processos” que simulam os trens executam como tarefas de fundo, sem qualquer interface de comunicação com o usuário, com um ciclo de trabalho de 100 ms. O funcionamento destes “processos” a cada ciclo consiste essencialmente em atualizar a posição do trem correspondente a partir da velocidade e do tempo decorrido desde a última atualização, sempre respeitando os procedimentos necessários para evitar situações catastróficas.

**3. Aspectos referentes aos trabalhos da 2ª e 3ª unidade**

O desenvolvimento deste trabalho servirá para a atribuição das notas da 2ª e da 3ª unidade. A única diferença é que, no caso da 3ª unidade, a comunicação entre o lançador e o gerenciador deverá ser feita através de *sockets* TCP/IP. Ou seja, poderá haver vários *softwares* lançadores em máquinas distintas, se comunicando com o gerenciador para realizar a atualização dos trens.

O encerramento do lançador em uma máquina específica só implicará na remoção dos trens que aquele lançador estava controlando. Dessa forma, os lançadores não poderão atualizar as informações ou remover trens lançados em máquinas diferentes.

**4. Forma de avaliação**

Será fornecida uma configuração (número de trens, números de vagões por trem, velocidades ...), com a qual os alunos deverão fazer a inicialização dos trens na malha ferroviária. Após introduzir a configuração no sistema, o grupo deverá deixar a aplicação executando por um tempo a ser determinado pelo professor.

**O grupo para o qual ocorrer alguma situação catastrófica terá nota 0,0 (zero)**. Para os demais grupos a nota será calculada a partir do percentual de velocidade máxima que foi conseguido para cada um dos trens: terá maior nota o grupo que atingir o maior percentual médio. **Também será atribuída nota 0,0 (zero) para o grupo que obtiver qualquer percentual maior que 100**%.

**5. Critérios de implementação**

Durante o desenvolvimento deste projeto, os alunos deverão obedecer aos seguintes critérios de implementação:

1. A aplicação deverá ser desenvolvida em Qt;
2. O aluno/grupo **poderá** utilizar o QtCreator;
3. A estrutura de pastas deverá ser semelhante àquela criada para os trabalhos anteriores;
4. Cada classe deverá ser implementada da mesma maneira que no exemplo desenvolvido em sala de aula, dividida em dois arquivos: classe.h e classe.cpp.
   1. No arquivo \*.h deverão estar contidos todos os comandos #include necessários para o desenvolvimento da classe bem como a definição da classe;
   2. No arquivo \*.cpp deverão ser implementados todos os métodos estabelecidos no arquivo \*.h;
   3. O arquivo \*.cpp, não deverá conter nenhum comando do tipo include, exceto aquele do respectivo arquivo \*.h;
5. Os arquivos \*.h e \*.cpp deverão conter as diretivas de pré-processamento #ifndef, #define e #endif, conforme exemplos desenvolvidos em sala de aula;
6. O arquivo do(s) projeto(s) deverá(ão) conter, além das configurações padrões, as seguintes linhas de código a serem inseridas ANTES da linha #Inputs:

CONFIG += release

MOC\_DIR += moc

OBJECTS\_DIR += obj

UI\_DIR += ui

1. A compilação do código não deve produzir erros;
2. Todas as bibliotecas utilizadas que não façam parte da API do Qt deverão ser previamente autorizadas pelo professor através de contato direto ou através do e-mail diogolr@gmail.com.

A desobediência de qualquer um dos critérios de implementação acima estabelecidos acarretará em imediata redução da nota em 50% de seu valor final.

**6. Pontuação extra**

Os aluno/grupos que desenvolverem seus projetos atendendo aos critérios abaixo listados poderão ter acrescidos à suas notas os respectivos valores representados entre colchetes.

* [1,0 ponto] Documentação completa de todas as classes desenvolvidas utilizando a ferramenta doxygen
* [0,5 ponto] A melhor interface gráfica, escolhida através de votação pelos alunos. Em caso de empate, o voto de minerva será dado pelo professor
* [0,5 ponto] Exibição dos estados dos semáforos utilizados no projeto

Para usufruir da pontuação extra, o grupo não poderá incorrer em nenhuma das situações descritas nos itens 1, 4 e 5 que venham a reduzir sua nota. A pontuação extra **é cumulativa** e **poderá** ser usada em benefício do aluno na pontuação da 1ª unidade, a critério do professor.