

Instituto Superior de Engenharia de Coimbra

Licenciatura em Engenharia Informática

Ano Letivo 2019/2020

**Trabalho Prático de Sistemas Operativos II**



**SOTáxi**

**Docente Responsável:**

Professor Francisco Pereira

**Trabalho realizado por:**

Nuno Simões, 2016017087

**Índice**

1. Estruturas de Dados...................................................................................

1.1. Centaxi..........................................................................................................3

1.2. Contaxi..........................................................................................................5

1.3. Conpass..........................................................................................................6

1.4. Mapinfo.........................................................................................................7

2. Mecanismos de Comunicação.......................................................................

2.1. Centaxi -> Contaxi..........................................................................................8

2.2. Contaxi -> Centaxi..........................................................................................9

2.3. Centaxi -> Conpass..........................................................................................9

2.4. Conpass -> Centaxi..........................................................................................9

2.5. Mapinfo -> Centaxi........................................................................................10

2.6. Centaxi -> Mapinfo........................................................................................10

3. Threads Usadas.........................................................................................

4. DLL...........................................................................................................

5. Interface Gráfica......................................................................................

6. Requisitos implementados.........................................................................

**Introdução**

Foi desenvolido um projecto em C para ser eexecutado em Windows. Utilizou o Visual Studio 2019.

A solução visual Studio é composta por 6 projectos que criam no total 4 aplicações disitintas que comunicam entre si utilizando mecanismos de *Inter Process Communications* (IPC). As aplicações simulam um sistema de controlo táxis. As aplicações desenvolvidas são:

* Central de táxi (CenTaxi) (só uma instância)
* Taxi (ConTaxi) (múltiplas instâncias)
* Passageiro (ConPass) (só uma instância)
* Representação gráfica do mapa da cidade (MapInfo) (só uma instância)

**Comunicação entre aplicações**

Para comunicar entre as várias aplicações foram utilizadso so seguinte mecasnismo de IPC:

* Named Pipes (ainda não implentado)
* Memória Partilhada
* Semáforos
* Eventos

Incluir aqui desenho

* 1. **ConTaxi -> CenTaxi**

Na comunicação entre a aplicação ConTaxi e a CenTaxi, no sentido ConTaxi para CenTaxi é feita exclusivamente com recurso a memória partilhada e semáforos, utilizando um mecanismo de produtor consumidor, num buffer ciruclar, em que ConTaxi é o produtor e CenTaxi é o consumidor. Podem existir várias aplicações ConTaxi a correr em simultâneo, em que cada uma representa um Taxi, com um Id distinto.

Com o mecanismo de produtor consumidor é possível vários ConTaxi enviarem pedidos em simultâneo para o CenTaxi e este conseguir processá-los 1 de cada vez.

A memória partilhada é utilizada para toda a comunicação entre o táxi (Contaxi) e a central (Centaxi). Nesta, são utilizados mecanismos de sincronização para o táxi saber onde deve escrever e ler os dados do buffer circular, baseando-se num método de comunicação entre processos Produtor-Consumidor.

Neste tipo de comunicação , é partilhado um buffer de tamanho fixo temporário, onde o Centaxi vai enviar pedidos/informações através do mesmo para a central e esta recolhe a informação, processando-a de seguida.

A utilização dos semáforos garante a exclusão mútua e a sequência correta na execução das operações, por isso o táxi (Produtor) apenas enviará dados para a central sempre que a central (Consumidor) estiver apta para os receber. Ou seja, espera que o consumidor retire elementos do buffer para que este não se encontre cheio.

É possível fazer um controlo preciso do buffer através de dois ponteiros: um aponta para a posição de leitura e outro para a de escrita de forma a saber quando é possível aceder ao buffer para ler ou escrever, respetivamente.

Da mesma forma, a central deve parar quando o buffer estiver vazio a fim de o táxi poder inserir dados no buffer.

Inicialmente, o táxi coloca uma mensagem do tipo “register” na zona de mensagens da memória partilhada na posição de escrita do buffer circular, e assim que a central detetar que existe uma nova mensagem, vai ler imediatamente a mensagem.

O Centaxi, através de uma thread “wait\_messages”, lê e trata a mensagem. Consoante o tipo, efetua uma ação, avisando o taxi através de um evento de que existem alterações e os táxis podem efetuar a leitura.

* 1. **Centaxi -> Contaxi**

Da central para o táxi existem dois tipos de comunicação distinhos:

* Memória partilhada: Como a central e o táxi utilizam a mesma zona de memória, é usado semáforos para efetuar a sincronização. Através do buffer circular anteriormente referido onde a central processa as mensagens na posição de leitura, retira-a do buffer e envia a informação atualizada para o táxi na posição de escrita.
* Named Pipes: Tratando-se de named pipes de uma só via, a central irá escrever na extrimidade do pipe e cada táxi terá uma struct para receber os dados na outra extremidade. Este mecanismo é utilizado para a central notificar o táxi de um passageiro atribuído ou que vai encerrar.
  1. Examplo comunicação ConTaxi para CenTaxi

Registo de um táxi: o taxi coloca uma mensagem do tipo “register” no buffer circular e fica à espera de uma resposta do CenTaxi (Há um semáforo) para cada um dos Taxis. A central lê a mensagem, trata-a e verifica se há espaço para um novo táxi no sistema. De seguida, enviaa reposta para o Taxi, utilizando uma memória partilhada exclusiva entre esse Taxi e a CenTaxi (conforme descrito em cima). Há um semáforo para indicar que a CenTaxi já enviou uma resposta ao Taxi.

* 1. **Centaxi -> Mapinfo**

A aplicação MapInfo só consome informação enviada pelo CenTaxi, utilizando memória partilhada para ir consultar informação do Mapa, Taxis e Passageiros. Para não estar constantemente a consultar a memória partilhada, é utilizando um Evento para notificar quando é necessário atualizar o mapa, por exemplo quando um táxi muda de posição ou um novo passageiro pede um serviço. Quando o MapInfo recebe o evento, força um update da janela com a informação do mapa.

* 1. **Centaxi -> Conpass**

Comunicação será feita por named pipes para a Meta 2.

* 1. **Conpass -> Centaxi**

Comunicação será feita por named pipes para a Meta 2.

1. **Aplicações / Bibliotecas**
   1. **Centaxi**

O CenTaxi guarda em memória informação sobre os passageiros, táxis e o mapa. O número máximo de passageiros e táxis podem ser definidos quando a aplicação inicia através de parâmetros da linha do comando. Caso não sejam definidos são utilizados os valores padrão 10 para ambos os casos.

Para alterar o núemro máximo de passeiros, utilizar o seguinte paramentro -p.

CenTaxi -p 5

Para alterar o número ma´ximo de táxis utilizar o seguinte paramentos -t.

CenTaxi -t 5

Também é possível definir qual o ficheiro do mapa a utilizar. Se não for definido será utilizado o ficherio map.txt. O formato deste ficheiro é com o caracter ‘X’ a indicar edificos e ‘ ‘ (espaço) a indicar as estradas. O fim de linhas do ficheiro deve ser apenas um ‘\n’. O ficheiro tem de ter no mínimo 50 colunas e 50 linhas. Para alterar o ficheiro a carrergar, utilizar o seguinte parâmetro -m.

CenTaxi -m newmap.txt

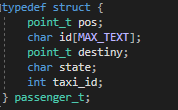
Este paramertos podem ser combinados e utilizados em conjunto e a ordem é irrelevante.

CenTaxi -t 5 -p 5 -m newmap.txt

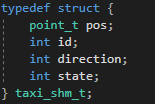
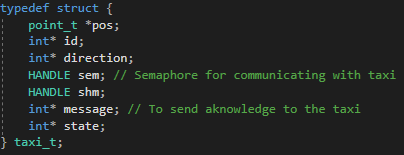
A informação guarda em memória parte dela é mapeada para memória partilhada para poder ser utilizada pelas outras aplicações (ConTaxi e MapInfo).

São criadas 6 zonas de memória partilhada.

* SHM VARIABLES: Onde é guarda a informação com o núremo de táxis e passageiros atuais, bem como o seu tamanho máximo. Esta informação é necesara para quando estiver a ser mapeados a SHM TAXI LIST e SHM PASSENGERS LIST.
* SHM TAXI LIST: Array com informação para guardar o número máximo de táxis definidos qual o CenTaxi iniciou.
* SHM PASSENGERS LIST: uigal ao de ciam
* SHM MAP SIZE: Onde está informação do taanho do map, para posteriormente ser mapeada SHM MAP com o tamanho correcto.
* SHM MAP: blblas
* SHM TAXI MESSAGES: Zona de memória do buffer circular utilizado para o mecanismo de comunicação entre ConTaxi e o CenTaxi.



Estrutura passenger\_t criada para guardar os dados de cadapassageiro. Um passeiro tem uma identificação única (id que é texto livre), um ponto de recolha (pos) e um ponto onde quer ir (destiny). O passeiro pode ter 3 estados: Sem Taxi atribuído, A aguardar táxi, dentro do táxi. Nos últimso 2 estados, é preenchido o táxi\_id com o ID do táxi que o irá buscar ou está a transportar.

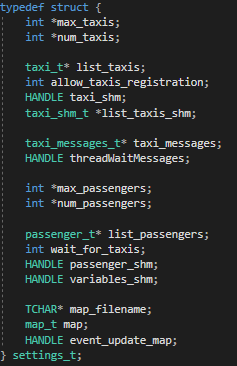
A estrutura tadxi\_shm\_t representa o Taxi na meória partilhada. Existe outra estrutura táxi\_t, que tem a mesma informação (ponteiros para todos as varia´veis da estrutura táxi\_t) além de informação da memória partilhada e semáforo do táxi, que é utilzaida para enviar mensagens de repostas aos pedidos do táxi.

Cada táxi tem uma posição (pos), um id único que é um numero (id), uam de 4 direções )direction) e o estado (livre, a ir buscar passageiro, a transportar um passeiro)

I<mg map\_t>, node\_t

Representa o mapa, tem um array dinâmico com os node\_t que representam cada quadricula do mapa. Cada node\_t tem corodenadsa X e Y e o tipo de quadricula (ediciio ou estrada).

Também tem a dimensão com 2 ponteiros para inteiros para memória partiljada SHM MAP SIZE.



CenTaxi tem uma estrutura settins\_t onde são galrados todos os dados, para evitar ´ter va´rias globais. Todos os dadso necessários são gaurdsos nesta esturura.

### Threads

Além da thrad principal que fica a aguarda comandos do telcado, há mais uma thread queestá a tratar de consumir nova mensagens de Taxis que sejam colocaso no buyffer icruclar da memória partilhasda.

* 1. **Contaxi**

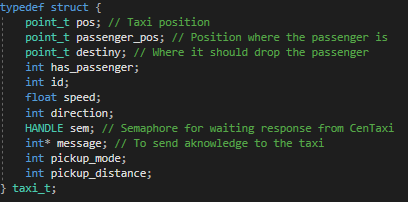
Aplicaçaõ para reprsentar um Taxi e que permite controlar o modo de funcionamento do táxi. Esta aplicação pode ser executada múltiplas vezes em simultâneo, sendo que cada uma deve representar um táxi com um ID distinto.

Para controloar o táxi há vários comandos de consola que podem ser executados

* Colcao alitas
* speedup: para acelarr o táxi até o ma´ximo de 3 quadriculas por senguido

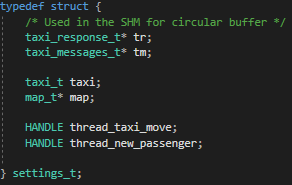
O Taxi desloca-se de modo autonomom na velocidade definida e em cada cruzamento escohe uma direcção aleatória, caso não esteja a ir buscar um passageiro (aidna não foi testado, por naõ ter sido implementado a aplicação ConPAss)

Ao inciar é criar meória partilhada e um sema´feor utilzaindo o ID do táxi, para garantir que são exclusivos deste táxi que serve para receber respostas do CenTaxi.



Estrutura criada para guardar os dados de cada **táxi** é mais completa do que na CenTaxi, pois o táxi tem mais informação que a central não tem. Além da informação descrita anteriormente, também gaurda a velcodidade (speed), raio para recolher passafeiros autoamticametne )pickup\_distance), se está em modo automático de recolha de passageiros (picp\_mode), se tem um passeiro (has\_passenger) e se tiver, etm a posição onde deve recoleher esse passagerior (passenger\_pos) e onde deve entregar o passeageiro (destiny).

O táxi também acesso à representação do mapa acedendo à memória partilhada do CenTaxi para o Mapa O ConTaxi . nunca tem acesso à lista de passageiros.



Como na CenTaxi, tabém foi criada uma estrurura settings\_T para guardar toda a informação necessária pelo táxi, para evitar o uso de varia´veis globais.

### Threads

Além da thrad principal que fica a aguarda comandos do telcado, há mais 2 duas threads. Uma para controlar o movimento automático do táxi. Outra para aguardar mensagens via named pipe do Cen

* 1. **Conpass**

Ainda está em desenvolvimento e naõ ainda nenhum utilizada prática.

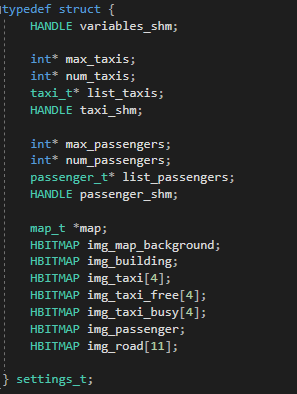
* 1. **Mapinfo**

O MapInfo é uam aplicação cujo obietivo é apenas mostrar graficamente o onde estão os táxis e os passageiros num mapa. Toda a informação é acedidad através de memória partilhada que foi alcoada anteriormente pelo CenTaxi.

Para suportar diferentes tipos de visualizaçõa do mapa foi impletamto um suporte para temas, em que devem existir 25 ficehiros BMP com nomes específicos para poder ter representações diferentes. Para já foram definidos 2 temas “default” e “blank”. Para alterar o tema deve ser utilizado o menu.

Ao escolher o tema, esta iformação fica guarda no Registry do Widnows no caminho Computer\HKEY\_CURRENT\_USER\Software\MapTaxi na chave TemplateFolder. Assim sempre que a aplicação iniciar, caso esteja definido um tempalte, este é carregado. A informação guardada é a pasta onde estão os BMP.

OS BMP são todos carregados no inicio ou ao mudar o tema e guardados em meoria para aceleara o processo de desenho do mapa na janela.

****

Como nas outas aplicações também foi criada uma estrutira settings\_t, mas dado o tempalte de aplicação Windows GUI, a estrutura é utilzaida numa variável global.

### Threads

Esta aplicação tem 2 threads. A principal onde é tratado o fluxo normal de uma palicação Windwos GUI. A outra thread fica a aguarda o Evento EVENT UPDATE MAP, que aos er recebido força uma atualização do mapa.

* 1. **ConTaxiDll**

Esta DLL foi criada com funções comuns em todos os projectos para facilitar a utilizaçºao dos IPC entre as ´varias aplicações. Cada projecto que utiliza esta DLL tem de a referenciar implifcitamente, no proejcto do Visual Sutido

* 1. **Console**

Esta DLL foi criada uma vez que há´3 apliacções que utilizam comandos da consola, e para simplificar o processo emc ada uma edlas que iria ser igual, excepto os comandos a serem executados.

Para utilizar a DLL tem de se referencidiar implicitamente e devem ser utilzidas as seguintes funções:

* init\_commands: Para iniciarlizar um array cque permite guardar os comandso que pode ser secutoadas
* register\_command: regista um comando (texto) que irá posteriormente executar uma função (que indicada como parâmetro). Também é indicado um texto de ajuda, para mostrar.
* read\_commad: Fica a aguarda por umcoamdno escrito no tecaldo.
* execute\_command: Tenta executar um comando lido do teclado. Caso seja válido executa-o, caso contrário mostra a lista de comandos disponíveis

Além destas funõse existem outras para simplificar o processo de ler informação dot ecladao:

* read\_number: ler um numero inteiro do teclado
* read\_text: ler texto do teclado
  1. **SO2\_TP\_DLL\_32**

Esta DLL foi fornecida para ser utilizada através da ligação explicita, pois não temos um ficheiro Lib, nem um ficheiro .h.,

Foi criada uma função para fazer o load e unload desta DLL no ContaxiDLL, pois é apra ser utilizada em todas as aplicações.

Após carregada manualmente e saõ procuradas as funções “dll\_register” e “dll\_log” de forma a registar todos os mecanismos de comunicação e sincronização e respetiva utilização.

Foram criadas duas várias globais com os ponteiros para estas funções

1. **Interface Gráfica**
2. **Requisitos Implementados**

|  |  |  |
| --- | --- | --- |
| **Num.** | **Requisito** | **Estado** |
|  |  |  |
|  |  |  |
|  |  |  |