U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Sistemas Operativos 2º Projeto

Estrutura de Mensagens trocadas

O cliente envia ao servidor uma mensagem contendo a sua PID, o número de lugares que quer reservar e uma lista com os lugares pelos quais tem preferência.

O servidor, respondendo, envia ao cliente uma mensagem com o número de lugares reservados em caso de sucesso ou com um código de erro (inteiro negativo) em caso de insucesso.

Request :

int time_out - Tempo limite de espera pela resposta do servidor
int num_wanted_seats - Número de lugares desejados pelo cliente
int pref_eat_list[MAX_SEAT_LIST] - Lista de lugares que o cliente prefere
<time_out> <num_wanted_seats> <pref_eat_list[MAX_SEAT_LIST]>

A request é enviada pelo cliente para o servidor através do FIFO requests.

Answer :

<Número de lugares reservados> <Lista de lugares reservados>

A answer é enviada pelo servidor para o respetivo cliente através do seu FIFO.

Carolina Azevedo | Daniela João | Pedro França | T7G02

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Mecanismo de Sincronização Utilizado

O mecanismo utilizado foi o mutex. Sendo este um *lockable object* projetado para sinalizar quando secções críticas do código precisam de acesso exclusivo, impedindo que outras *threads* com a mesma proteção sejam executados simultaneamente e acedam aos mesmos locais de memória.

Neste caso, trata-se do acesso ao array de lugares disponíveis para venda. Desta forma, cada bilheteira acede a um lugar não deixando as outras aceder, por consequência, evitamos casos como um lugar ser reservado simultaneamente para clientes diferentes.

Encerramento do Servidor

É usado o sinal de alarme SIGALRM para contar o tempo especificado na função open_time() das bilheteiras do servidor.
Quando este alarme dispara (passado o tempo especificado) este sinal é "apanhado" pelo respetivo *handler* que ativa uma *flag*, a qual após a ativação, faz as *threads* das bilheteiras saírem do seu ciclo *while* e terminar.
Antes de terminar a *main thread,* é feito um pthread_join() com o intuito a esperar que todas as *threads* de bilheteiras terminem.