Universidade do Minho
Escola de Engenharia
Licenciatura em Engenharia Informática

Unidade Curricular de
Sistemas Distribuídos

Ano Letivo de 2022/2023

Trabalho Prático
Gestão de frota

Grupo 21
a83630 - Duarte Serrão
a95125 - Hugo Martins
a96075 - João Escudeiro
a84696 - Renato Gomes

8 de janeiro de 2023

Índice

1 Introdução **1**

1.1 Contextualização 1

2 Protocolo **1**

2.1 Cenários 1

2.2 Tipos de Mensagens 2

3 Estrutura do Programa **3**

3.1 Cliente 3

3.2 Servidor 3

3.2.1 Estruturas criadas 3

3.2.2 Conexão com cliente 4

3.2.3 Recompensas 4

3.2.4 Exclusão mútua 4

4 Conclusão e Trabalho Futuro **4**

5 Anexos **5**

5.1 Cenários 5

1 Introdução

1.1 Contextualização

Como componente prática da Unidade Curricular de Sistemas Distribuídos 22/23 , surge um trabalho que consiste na gestão de um Frota de Trotinetes. Como componentes principais ter-se-á um (ou vários) clientes e um servidor que serve de "Handler"para todos os pedidos dos clientes. Um cliente pode pedir as trotinetes mais proximas , obter recompensas, reservar trotinete ou iniciar viagem .O servidor , tendo conhecimento do mapa , responde aos pedidos , bem como calcula as recompensas enviando-as através de um sistema de notificações.

2 Protocolo

Para que seja possível o cliente e o servidor comunicarem de forma eficaz, foi preciso criar um protocolo que define certos pedidos ou respostas entre as duas entidades. Em primeiro lugar, observou-se os cenários expostos pelo enunciado disposto pelos docentes e de seguida gerou-se um conjunto de tipos de mensagens.

2.1 Cenários

Tendo por referência o enunciado disponibilizado pelos docentes da unidade curricular de Sistemas Distríbuidos, é possível gerar alguns cenários do programa.

Pedido	Cenários Gerados	Fig.
Autenticação e registo de utilizador, dado o seu nome e palavra-passe. Sempre que um utilizador desejar interagir com o serviço deverá estabelecer uma conexão e ser autenticado pelo servidor.	• Registar	1
	• Login	2
	• Logout	3
Listagem dos locais onde existem trotinetes livres, até uma distância fixa D de um determinado local.	• Trotinetes perto	4
Listagem das recompensas com origem até uma distância fixa D de um determinado local, dada por pares origem-destino	• Recompensas perto	5
Reserva de uma trotinete livre, o mais perto possível de determinado local, limitado uma distância fixa D. O servidor deverá responder com o local e um código de reserva, ou código de insucesso, caso tal não seja possível.	• Reserva de trotinetes	6
	• Começar viagem	7

Pedido	Cenários Gerados	Fig.
Estacionamento de uma trotinete dando o código de reserva e o local. O servidor deve informar o cliente do custo da viagem, em função do tempo passado desde a reserva e da distância percorrida. Caso a viagem corresponda a uma recompensa, é informado do valor da recompensa. A aplicabilidade da recompensa deve ser avaliada no estacionamento, de acordo com a lista de recompensas em vigor nesse momento.	• Fim da viagem	8
Um cliente pedir para ser notificado quando apareçam recompensas com origem a menos de uma distância fixa D de determinado local. As notificações poderão ser enviadas muito mais tarde,devendo entretanto o cliente poder prosseguir com outras operações. O cliente poderá cancelar os pedidos de notificação.	• Toggle receber notificações	9

2.2 Tipos de Mensagens

- `GENERIC`: Mensagem genérica contendo apenas um bloco de texto.
- `REGISTER`: Contem um username e uma password de um utilizador novo.
- `CONNECTION`: Contem um username e uma password de um utilizador existente.
- `CONNECTION_RESPONSE`: Sucesso da autenticação do cliente.
- `DESCONNECTION`: Contem um username.
- `DESCONNECTION_RESPONSE`: Sucesso do logout do cliente.
- `NEARBY_SCOOTERS`: Pede uma lista de trotinetes que estão até uma certa distância.
- `LIST_SCOOTERS`: Resposta ao cliente com a lista de trotinetes.
- `NEARBY_REWARDS`: Pede uma lista de recompensas que estão até uma certa distância.
- `LIST_REWARDS`: Resposta ao cliente com a lista de recompensas.
- `SCOOTER_RESERVATION` Mal o cliente pede para reservar uma trotinete dando a sua localização.
- `START_TRIP`: Cliente pede para começar a viagem, dando o seu código de reserva.
- `SCOOTER_RESERVATION_RESPONSE`: Mal o cliente pede para reservar uma trotinete, o servidor responde com um código de reserva.
- `END_TRIP`: Cliente pede para acabar a viagem, dando a sua localização e o código de reserva.

- `COST_REWARD`: Resposta ao cliente com o custo e os créditos ganhos devido a recompensas.
- `TOGGLE_NOTIFICATIONS`: Cliente pede para ligar ou desligar as notificações.

3 Estrutura do Programa

uma conexão por cliente por tcp

3.1 Cliente

Quando iniciamos um cliente novo, este tenta estabelecer rapidamente uma conexão através de um socket TCP/IP na porta 4999 do servidor. A partir deste momento , e com a ajuda da interface de um menu, o cliente tem à sua disposição 5 opções, sendo elas : obter as trotinetes mais próximas, obter as recompensas disponíveis, reservar trotinete, ligar notificações e dar logout. Todas as opções realizarão Queries ao servidor, para as quais, o cliente ficará à escuta para tentar obter uma resposta.

As mensagens antes de ser enviadas são serializadas numa classe auxiliar. A partir do momento que receber um socket de resposta do servidor, o cliente começará uma thread nova que irá fazer o processamento do socket, gerando assim a resposta correta.A mensagem recebida é deserializada , para se conseguir obter o conteúdo da mesma num formato apelativo(texto). O uso de multithreads nas respostas deveu-se essencialmente á necessidade que o grupo sentiu aquando a realização das notificações.

3.2 Servidor

3.2.1 Estruturas criadas

Para que o servidor se mantenha a par do estado da alicação, foram criadas 4 estruturas novas, sendo que uma é para lidar com as trotinetes, outra para reservas e as outras duas para clientes.

- `trotinetes`: Lista de localizações com o número de trotinetes em cada;
- `contasAtivas`: Um mapa que liga um port, ou seja, identificaddor de um cliente a um objeto do tipo Utilizador;
- `notificationBros`: Uma lista de ports, ou seja, de clientes, que desejam receber notificações;
- `reservasAtivas`: Um mapa que liga um código de reserva a um objeto do tipo Reservation.

3.2.2 Conexão com cliente

O servidor dispõe de uma thread cujo principal objetivo é ouvir pedidos de vários clientes. Nesta thread , para cada conexão nova será gerada uma thread auxiliar que tratará de desserializar o pedido , e posteriormente gerar a resposta ao mesmo. Existem algumas funções que são capazes de processar os diferentes pedidos como por exemplo obter a lista de trotinetes mais próximas de uma dada localização , iniciar viagem , gerar preço da viagem , entre outras. Esta resposta será serializada e enviada de volta para o cliente correto. Isto acontece pois na criação da thread que trata os pedidos é passada a referência para o socket daquele cliente , garantindo assim que as respostas certas são enviadas para os clientes certos.

3.2.3 Recompensas

Quanto às recompensas , as mesmas são calculadas da seguinte forma: quando na posição indicada pelo cliente existem duas ou mais trotinetes, o servidor guarda as posições todas do mapa com 0 trotinetes e aplicando-lhes um filtro. Esse filtro permite guardar apenas as posições sem trotinetes e que num raio D não existem posições com trotinetes. Para cada uma dessas posições será uma recompensa esse cálculo é tanto maior quanto a distância entre a posição origem e a posição destino.Sempre que um cliente tem as notificações ativas e insere uma localização , o sistema notifica o cliente sobre as trotinetes proximas com recompensa.

3.2.4 Exclusão mútua

Sempre que usamos threads em paralelo houve a necessidade de garantir a exclusão mútua e a proteção dos dados. Assim para cada região crítica foram criados e implementados locks, garantindo assim que nunca duas threads usariam a mesma região crítica ao mesmo tempo.

4 Conclusão e Trabalho Futuro

Dado por concluído a componente prática do trabalho , o grupo autoavalia-se de forma bastante positiva visto que todos os requisitos mencionados no trabalho foram cumpridos . No geral , as tarefas mais difíceis foram a implementação de um sistema de recompensas inteligente o suficiente para conseguir calcular as recompensas a partir de um local A para um local B , e também a implementação do servidor , na parte das mensagens , pois ocorriam alguns bugs na troca de mensagens , dado que o cliente é multithread , e o servidor também.Caso houvesse mais tempo , haveria alguns aspetos que podiam ser melhorados , para aumentar um pouco a eficiência do programa como por exemplo alguns locks dados no servidor serem alterados .No geral o grupo conseguiu implementar todas as funcionalidades impostas no trabalho.

5 Anexos

5.1 Cenários

Registar

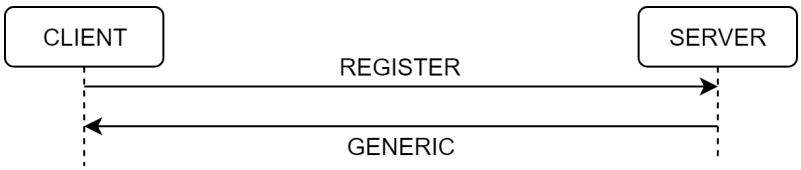

Figura 1: Cenário "Registar".

Login

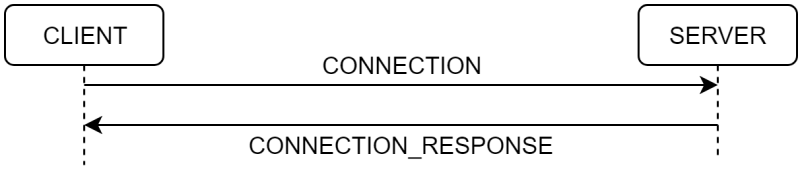

Figura 2: Cenário "Login".

Logout

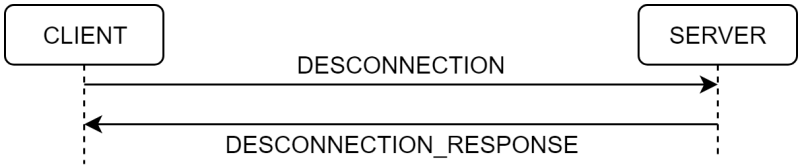

Figura 3: Cenário "Logout".

Trotinetes perto

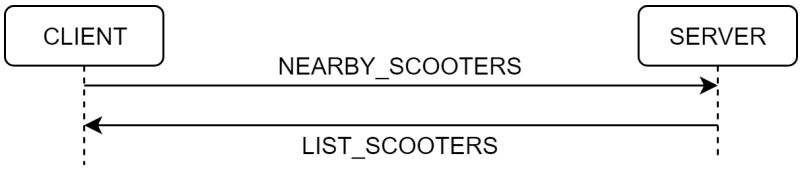

Figura 4: Cenário "Trotinetes Perto".

Recompensas perto

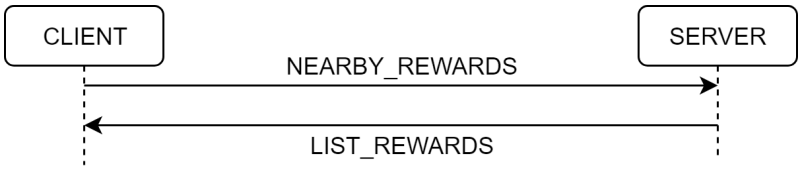

Figura 5: Cenário "Recompensas Perto".

Reservar trotinete

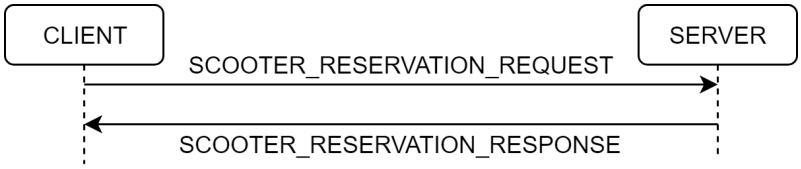

Figura 6: Cenário "Reservar trotinete".

Começar viagem

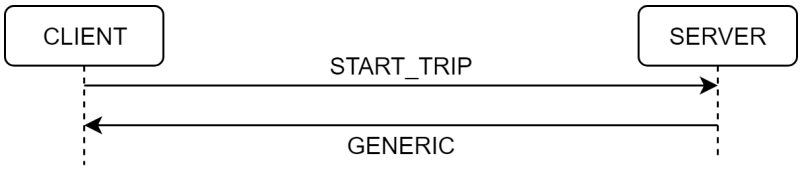

Figura 7: Cenário "Começar viagem".

Acabar viagem

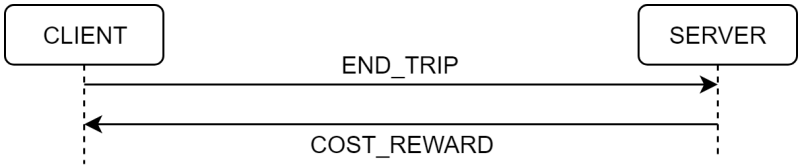

Figura 8: Cenário "Acabar viagem".

Toggle receber notificações

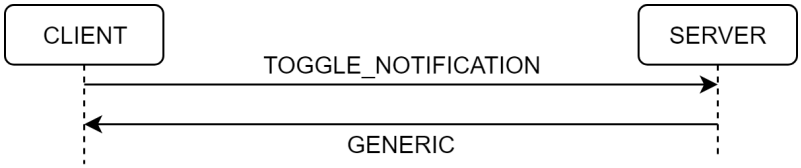

Figura 9: Cenário "Toggle receber notificações".