Sistemas Distribuídos — Trabalho Prático
Gestão de frota

Engenharia Informática
Universidade do Minho

2022/2023

Informações gerais

- Cada grupo deve ser constituído por 4 elementos, obrigatoriamente inscritos no *eLearning* até 18 de novembro.
- Deve ser entregue o código fonte e um relatório de até 6 páginas (A4, 11pt) no formato PDF.
- O trabalho deve ser entregue até às 23:59 do dia 9 de janeiro de 2023 no *eLearning*.
- A apresentação do trabalho será em dia e hora a agendar posteriormente por cada grupo.
- Cada grupo deve organizar a sua apresentação de forma a que todos os elementos participem espontaneamente.

Resumo

Neste projeto pede-se a implementação de uma plataforma de gestão de uma frota de trotinetes elétricas, sob a forma de um par cliente-servidor em Java utilizando *sockets* e *threads*.

A essência do serviço é permitir aos utilizadores reservar e estacionar trotinetes em diferentes locais. O mapa é uma grelha de $N \times N$ locais, e.g., $N = 20$, sendo as coordenadas geográficas pares discretos de índices. Para a distância entre dois locais considera-se a norma l^1, ou seja a distância Manhattan.

Para levar a uma boa distribuição das trotinetes pelo mapa, existe um sistema de recompensas, em que utilizadores são premiados por levaram trotinetes estacionadas num local A para um local B. Em cada momento existe uma lista de recompensas, atualizada ao longo do tempo, sendo cada recompensa identificada por um par origem-destino, com um valor de recompensa associado, calculado em função da distância a percorrer.

Funcionalidade

Este serviço deverá suportar a seguinte funcionalidade:

1. 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.
2. Listagem dos locais onde existem trotinetes livres, até uma distância fixa D (e.g., $D = 2$) de um determinado local.
3. Listagem das recompensas com origem até uma distância fixa D (e.g., $D = 2$) de um determinado local, dada por pares origem-destino.

4. 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.

5. 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.

6. 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.

Geração de recompensas

O sistema de recompensas deve correr em *background*, avaliando a distribuição de trotinetes livres pelo mapa quando se justifique. Concretamente, deve gerar uma recompensa para movimentar uma trotinete de um dado local A para um local B quando: há mais do que uma trotinete livre em A e não há nenhuma livre num raio D de B.

O sistema deve reavaliar a situação sempre que e apenas quando alguma trotinete livre é reservada ou estacionada. Assuma uma distribuição aleatória de uma dado número fixo de trotinetes pelo mapa, todas livres, quando o servidor arranca.

Servidor

O servidor deverá ser escrito em Java, usando *threads* e *sockets* TCP, mantendo em memória a informação relevante para suportar as funcionalidades acima descritas, receber conexões e input dos clientes, bem como fazer chegar a estes a informação pretendida.

Biblioteca do cliente

Deverá ser disponibilizada uma biblioteca (conjunto de classes e interfaces) que proporcione o acesso à funcionalidade do servidor descrita acima. Esta biblioteca deve ser independente da interface com o utilizador e deverá ser escrita em Java usando *threads* e *sockets* TCP.

Interface do utilizador

Finalmente, deverá ser disponibilizada uma interface com o utilizador que permita interagir com o serviço através da biblioteca cliente. Esta interface deverá também ser escrita em Java e tem como único objetivo a interação com o serviço para testes e durante a apresentação do trabalho.

Requisitos

Na implementação devem ser satisfeitos os seguintes requisitos:

- Cada cliente apenas pode estabelecer uma única conexão para o servidor, por onde pedidos, respostas e notificações devem passar.

- O protocolo de comunicação entre cliente e servidor deverá ser num formato binário, através de código desenvolvido no trabalho, podendo recorrer apenas a Data[Input|Output]Stream.
- Para o serviço não ficar vulnerável a clientes lentos, não deverá ter *threads* do servidor a escrever em mais do que um socket, devendo as escritas ser feitas por threads associadas a esse socket.
- Para manter os clientes responsivos, o sistema que gera recompensas deve correr no servidor em *thread(s)* específica(s), e nunca em *threads* associadas a clientes.

Valorizam-se estratégias que diminuam a contenção e minimizem o número de threads acordadas.