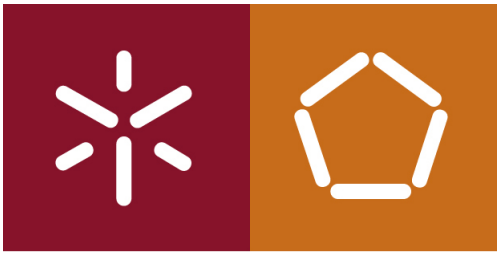

Universidade do Minho

Mestrado Integrado de Engenharia Informática

Análise e Teste de Software

Trabalho Prático 1

Diogo Soares	A74478
Ricardo Leal	A75278
Samuel Martins	PG37163

15 de Outubro de 2018

Conteúdo

1 Introdução **2**

2 Levantamento e Análise de Requisitos **3**

3 Teste a Grande Escala **4**
3.1 Ficheiro log e seu Parser 4
3.2 Resultados 5

4 Avaliação da Aplicação **6**

5 Conclusão **7**

1 Introdução

Neste primeiro projeto da unidade curricular de Análise e Teste de Software é pretendido que testemos um software desenvolvido por alunos no seu primeiro curso de programação em JAVA. Assim, numa fase inicial pretende-se verificar se o software cumpre os requisitos necessários definidos no seu enunciado. Para tal, criamos um *parser* ANTLR para povoarmos a aplicação com um grande número de dados, verificando assim se esta os consegue suportar e realizar a função pela qual foi implementada.

2 Levantamento e Análise de Requisitos

Numa primeira fase, tivemos de fazer o levantamento de requisitos propostas no enunciado para a aplicação da Umer. Em relação ao **Cliente**, este tem que poder:

- solicitar uma viagem ao táxi mais próximo das suas coordenadas;
- solicitar uma viagem a um táxi especifico;
- fazer uma reserva para um táxi especifico que, de momento, não está disponível.

Usando a aplicação, facilmente conseguimos concluir que a aplicação é capaz de realizar estas operações.

Quanto ao **Motorista**, estes têm de poder:

- sinalizar que estão disponíveis para serem requisitados;
- registar uma viagem para um determinado cliente;
- registar o preço que custou determinada viagem.

Enquanto eles podem alterar a sua disponibilidade, não podem registar viagens nem o preço destas, visto que a aplicação regista isso automaticamente.

O modo operacional da aplicação teria de ser o seguinte:

1. cliente indica onde se encontra;
2. escolhe que serviço quer utilizar;
3. táxi indica ao cliente o custo e o tempo previsto de viagem;
4. é calculado valor real da viagem com o valor estimado, ajustando-se o valor de pagamento;
5. táxi fica no destino;
6. cliente pode avaliar serviço.

O que verificamos ao testar a aplicação é que esta cumpre os requisitos impostos de utilização excetuando o ponto 3, onde o custo e o tempo de viagem são apresentados apenas no final do serviço.

Além disso também constatamos o bom funcionamento das empresas de táxi assim como nas filas de espera, onde os clientes ficam à espera que o seu condutor fique disponível.

Podemos também aferir que a aplicação cumpre os **requisitos** básicos:

- registar um utilizador, quer cliente quer motorista;
- validar o acesso à aplicação utilizando as credenciais;
- criar e inserir viaturas;
- associar motoristas a viaturas;
- solicitar, por parte de um cliente, uma viagem de ponto P(x,y) para o ponto R(x,y), escolhendo uma viatura ou então solicitando a viatura mais próxima. Caso a viatura seja das que possuem lista de espera, inserir na lista de espera;
- classificar o motorista, após a viagem;
- ter acesso, no perfil de cliente, à listagem das viagens efetuadas (entre datas);
- ter acesso, no perfil de motorista, à listagem das viagens efetuadas (entre datas);
- indicar o total faturado por uma viatura, ou empresa de táxis, num determinado período;
- determinar a listagens dos 10 clientes que mais gastam;
- determinar a listagem dos 5 motoristas que apresentam mais desvios entre o valores previstos para as viagens e o valor final faturado;
- gravar o estado da aplicação em cheiro, para que seja possível retomar mais tarde a execução,

3 Teste a Grande Escala

Para testar-mos o programa a grande escala foi-nos passado um ficheiro "log", que contem varias descrições de funções que o programa seria capaz de executar. Para traduzir-mos esse ficheiro para funções na aplicação, recorremos a um parser antlr.

3.1 Ficheiro log e seu Parser

No "log"existem oito tipo diferentes de "frases"em todo o ficheiro. Para cada frase criamos uma regra correspondente.

- **registar condutor:** para cada frase que se inicia com este prefixo, registamos um condutor com os dados passados pelo ficheiro. Se este não tiver indicação de pertencer a uma empresa, registamos também aquele que será o seu veiculo pessoal (de modo a podermos testar a aplicação);

- **registar cliente:** registamos o cliente e atualizamos as suas coordenadas de acordo os parâmetros passados pelo log;
- **registar empresa:** além de registarmos a empresa, registamos também cinco veículos na mesma (por razões de teste de aplicação) ;
- **login:** apenas guardamos numa variável global quem iniciaria sessão de modo a nas próximas ações saber quem as executaria;
- **logout:** variável global que contém *e-mail* do utilizador fica a *null*;
- **solicitar:** como esta frase só é acompanhada de coordenadas, a solicitação que decidimos por como *standard* foi a de solicitar ao táxi mais próximo. Assim, o cliente autenticado solicita ao táxi mais próximo disponível uma viagem para as coordenadas em questão;
- **viajar** e **recusar viagem:** decidimos ignorar estas frases na nossa gramática pois a aplicação não tem implementada essas opções para o condutor, visto que uma vez solicitada uma viagem por parte de um cliente, esta é realizada de forma imediata.

3.2 Resultados

Para verificarmos se o *parser* tinha introduzida toda a informação na aplicação e se esta deu uma boa resposta ao grande numero de dados, contamos quantas vezes as ações "principais"são chamadas no ficheiro log.

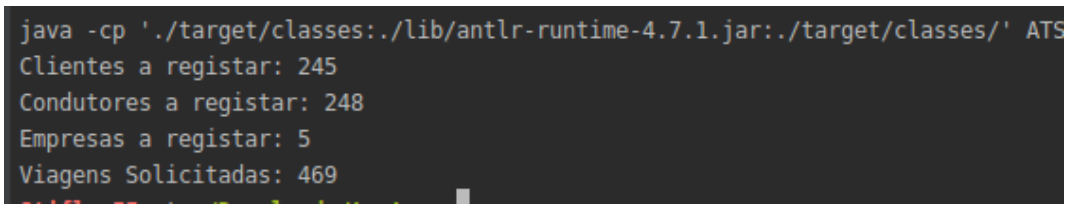

Figura 1: Número de funções a executar

Uma vez executado o *parser*, abrimos a aplicação com o *admin* verificando os dados presentes na aplicação e que as viagens, de facto acontecem.

Figura 2: Dados da aplicação

Verificamos assim, que esta deu uma resposta afirmativa à execução do *parser*. Uma vez na aplicação, fomos apurar os tops pedidos no enunciado.

Figura 3: Top 10 de Clientes

Figura 4: Top 10 de Condutores

Figura 5: Top 5 de Maiores Desvios

4 Avaliação da Aplicação

Uma vez concluídos todos os testes à aplicação, podemos aferir que esta cumpre os requisitos propostos no enunciado, focando se principalmente na parte do cliente. O condutor, por outro lado, não pode aceitar ou recusar serviço, o que quer dizer que se estiver disponível e este for o condutor mais próximo (caso o cliente escolha

esta opção) ou o condutor escolhido, este realizará a viagem pedida. O aspeto mais negativo que encontramos enquanto testava-mos esta aplicação, foi métodos lógicos presentes na interface.

5 Conclusão

Em suma, este trabalho permitiu nos estar pela primeira vez no lado do avaliador e permitir assim, que compreendamos as dificuldades de testar e avaliar um projeto desconhecido. Acreditamos que isto nos irá permitir melhorar também a nossa maneira de escrever um programa devido a esta nova consciência que ganhamos e iremos ganhar ao longo do semestre nesta UC.