Análise e Teste de Software

2018-2019

Projeto: UMeR

No contexto da disciplina de Programação Orientada a Objectos, no ano lectivo 2016-2017 foi proposto aos alunos criar um serviço de transporte de passageiros que faz concorrência a um serviço muito conhecido (e que tem um nome muito parecido com UMeR…) O enunciado desse projecto descreve informalmente os requisitos funcionais do sistema de software que os alunos tiveram de desenvolver, usando a linguagem de programação Java.

Uma possível solução para esse projeto é disponibilizada com este enunciado. De facto, essa solução foi desenvolvida por um grupo de alunos que frequentou a disciplina nesse ano lectivo, e foi avaliada pela equipa docente com a classificação de *Bom.* Esta avaliação foi feita com base em vários factores, incluindo a análise do funcionamento do sistema de software, demonstrado pelos alunos. Porém, estas demonstrações são muitas vezes feitas com poucos dados - não mostrando uma execução real da aplicação! - uma vez que não é possível no contexto de uma disciplina com um grande número de alunos/projeto testar devidamente este software.

Análise e teste da Aplicação UMer

Neste projecto de Análise e Teste de Software, pretendemos analisar e testar com grande detalhe o software desenvolvido pelos alunos. Assim, serão usadas técnicas e ferramentas que permitirão aferir da qualidade da aplicação desenvolvida pelos alunos.

O projecto deve ser realizado por grupos de 3 alunos e tem um peso de 50% da nota da disciplina.

1ª Fase: Validação da Solução dos Alunos de POO

O projecto será feito em três fases. Na primeira fase os grupos devem avaliar a solução apresentada pelos alunos de POO. Isto é, devem verificar se o software cumpre os requisitos definidos no seu enunciado. A solução deve ainda ser executada utilizando uma grande volume de informação. Assim, disponibiliza-se um ficheiro de *log* que contém o registo de todas as operações realizadas pela UMeR durante um certo período de tempo. Obviamente, a aplicação deve executar correctamente ao efectuar todas essas operações.

De notar que o ficheiro de *log d*efine uma linguagem (de domínio específico), que deve ser processada por um parser. Para desenvolver este parser, os alunos deverão utilizar o sistema ANTLR (www.antlr.org/): um gerador de parsers para Java. O parser deve ser integrado no software da UMer de modo a executar todas as operaçőes em *batch.*

2ª Fase: Teste do Software UMeR

Enquanto que a 1ª Fase incidia sobre a funcionalidade do projecto UMeR a nível do seu uso por um utilizador, nesta 2ª Fase pretende-se avaliar a tanto a funcionalidade do código como das funcionalidades disponíveis aos utilizadores. Desta forma, será necessário trabalhar sob dois aspectos:

- Testes unitários;
- Testes de sistema.

2.1: Testes unitários

O código de cada classe poderá ter erros. Implemente testes que permitem encontrar os erros eventualmente presentes do programa e tome nota caso detecte algum erro. Na definição de testes unitários tenha em atenção os critérios de qualidade de testes definidos nas aula teóricas (*critérios cobertura e uso de mutação de software em testing*).

2.1.1: Automação da Geração dos Testes Unitários

Utilize o sistema EvoSuite de modo a gerar testes unitários para a aplicação UMeR.

2.2: Testes de sistema

À semelhança da 1ª Fase, para testar o sistema será necessário executar instruções como se um utilizador as executasse. Para isso, será necessário gerar *inputs* aleatórios – semelhantes aos ficheiros de *log* que foram dados na 1ª Fase – e possivelmente com elementos adicionais para testar funcionalidades que os ficheiros de *log* fornecidos não permitiam.
Para realizar esta tarefa, terá de gerar novos casos de teste (*inputs* de *log*) e executá-los. *Recomenda-se a utilização do sistema QuickCheck.* Poderá ser necessário a actualização da gramática da 1ª Fase.

Resultado esperado

Como resultado desta fase, pretende-se que a framework de teste da aplicação UMer, que terão de desenvolver neste fase, produza um relatório que atesta a qualidade dos testes e da aplicação em causa.

Indique no relatório que documenta esta fase, que partes do código tem erros e como os encontraram, entre outros pontos que achem de relevo. É pretendido também saber que alterações fizeram ao código, tanto ao da 1ª Fase como ao do projecto UMeR. Estão no papel de *testers* com o objectivo de informar os *developers* de erros.

3ª Fase: Análise da Qualidade do Software UMer

A definir após terminar a 2ª Fase.

Avaliação:

A avaliação do projeto terá em conta os seguintes pontos:

- Qualidade das soluções entregues em cada uma das 3 fases de entrega;
- Qualidade da solução final;
- Uso do sistema de gestão de versões;
  (acesso a um sistema de controlo de versões será disponibilizado)
- Apresentação Final;
- Relatório Final.

Datas de Entrega:

1ª Fase: 09 de Outubro, 2018 (estendido para 16 de Outubro)
2ª Fase: 13 de Novembro, 2018 (estendido para 19 de Novembro)
3ª Fase: 18 de Dezembro, 2018
Apresentação Final: 08 de Janeiro, 2019.

POO (MiEI/LCC)

2016/2017

Enunciado do Trabalho Prático

Conteúdo

1 UMeR

Uma empresa de alunos de POO pretende criar um serviço de transporte de passageiros que faça concorrência a um serviço muito conhecido (e que tem um nome muito parecido com UMeR...). Pretende-se que a aplicação a ser desenvolvida dê suporte a toda a funcionalidade que permita que um utilizador realize uma viagem num dos táxis da UMeR. O processo deve abranger todos os mecanismos de criação de utilizadores, motoristas, automóveis e posteriormente a marcação das viagens, a realização das mesmas e respectiva imputação do preço. Pretende-se também que o sistema guarde registo de todas as operações efectuadas e que depois tenha mecanismos para as disponibilizar (exemplo: viagens de um utilizador, extracto de viagens de um taxi num determinado período, valor facturado por um taxi num determinado período, etc.).

Cada perfil de utilizador deve apenas conseguir aceder às informações e funcionalidades respectivas.

- Os clientes dos táxis UMeR poderão:
  - solicitar uma viagem ao táxi mais próximo das suas coordenadas;
  - solicitar uma viagem a um táxi específico;
  - fazer uma reserva para um táxi específico que, de momento, não está disponível.
- Os motoristas poderão:
  - sinalizar que estão disponíveis para serem requisitados;
  - registar uma viagem para um determinado cliente;
  - registar o preço que custou determinada viagem.

2 Actores do sistema

Propõe-se a existência de dois tipos distintos de actores no sistema, que partilham a seguinte informação:

- email (que identifica o utilizador);
- nome;
- password;
- morada;
- data de nascimento.

Para além disso, cada actor tem um conjunto de dados específicos, como explicado de seguida.

2.1 Cliente

O Cliente representa a pessoa que solicita e efectua uma viagem de táxi. O cliente está sempre numa determinada localização (expressa em x e y, isto é, num espaço 2D) e escolhe um táxi específico ou então solicita o táxi mais perto que esteja disponível.

O cliente tem também uma relação de todas as viagens que fez, com toda a informação relativa à viagem.

2.2 motorista - colaborador da UMeR

O motorista conduz o táxi e além da informação atrás referida tem também dados relativos a:

- grau de cumprimento de horário estabelecido com o cliente, dado por um factor entre 0 e 100;
- classificação do motorista, dado numa escala de 0 a 100, calculada com base na classificação dada pelo cliente no final da viagem;
- histórico das viagens realizadas;
- número de kms já realizados na UMeR;
- informação sobre se está ou não disponível em determindado momento, isto é, se está ou não a trabalhar.

3 Os táxis UMeR

Além dos clientes e dos motoristas, o ecossistema do UMeR contempla diferentes tipos de viaturas de aluguer (táxis). Neste momento estão em funcionamento os seguintes tipos de viaturas:

- carros ligeiros;
- carrinhas de nove lugares;
- motos.

Mas a gestão da UMeR considera que em qualquer altura pode aumentar o tipo de táxis que fazem parte da rede (neste momento está em estudo a utilização de bicicletas do tipo tandem!).

Cada um destes tipos de viaturas tem associada:

1. uma velocidade média por km;
2. um preço base por km;
3. um factor de fiabilidade, que determina a capacidade da viatura cumprir o tempo acordado com o cliente. Sempre que se realiza uma viagem é calculado (através da invocação de um random()) a capacidade de o veículo cumprir com o tempo acordado com o cliente. Este factor tem um efeito multiplicador sobre o tempo fornecido ao cliente.

Existem ainda alguns tipos de viaturas que possibilitam a existência de uma fila de espera de marcações. Quando o táxi não está disponível (por exemplo, pelo facto do condutor estar fora do horário de trabalho) é possível para essas viaturas aceitarem reservas de clientes. As reservas serão satisfeitas por ordem de chegada.

Uma viatura sabe sempre a localização (em x e y) onde está. Quando realiza um serviço desloca-se para as coordenadas indicadas pelo cliente e fica aí parado até que seja solicitado um novo serviço.

4 Fazer uma viagem no UMeR

O processo de fazer uma viagem no UMeR segue as seguintes regras:

1. o cliente indica as coordenadas x e y em que se encontra;
2. o cliente decide se pretende chamar um táxi específico ou então solicitar o que está mais próximo;
3. por uma questão de simplificação, os UMeR deslocam-se sempre em linha recta pelo que o cálculo da distância entre o cliente e o táxi é feito pela cálculo da distância euclidiana (exemplo: se o cliente estiver em (0,0) e o táxi em (2,2) a distância é de 2.8284 kms);
4. após ser calculada a distância consegue-se saber, dadas as características do táxi, quanto tempo demora a chegar ao cliente e depois ao destino que o cliente solicita;
5. o táxi indica ao cliente qual o custo estimado da viagem, tendo em conta o deslocamento que é necessário efectuar, e o tempo total de viagem;
6. de acordo com a fiabilidade do carro (e de outros factores que pode considerar: a destreza do condutor, as condições metereológicas, etc.) é calculado o tempo real da viagem. Se a diferença for superior a 25% do tempo estimado, então o preço a cobrar é o combinado com o cliente. Se a diferença for igual ou inferior a 25% o valor é ajustado para o valor real em função do tempo decorrido;
7. o táxi fica no ponto definido como fim da viagem à espera de nova solicitação de serviço;
8. após a viagem o cliente pode dar uma nota ao motorista e fica com o documento relativo à viagem guardado na sua área pessoal.

Como factor de diferenciação na solução que cada grupo de POO vai encontrar para resolver este problema, podem pensar que o tempo de viagem (logo o custo) além de ser função de um factor associado ao veículo, pode também ser função das condições atmosféricas ou condicionantes de trânsito. Nesse caso será necessário acrescentar informação a passar nos métodos que suportem estes requisitos.

5 Motoristas individuais vs Empresas de Táxi

Numa primeira fase o UMeR foi pensado para condutores que conduziam a sua viatura e fazem serviço de transporte de clientes. No entanto, e devido ao sucesso do negócio, foram criadas empresas que possuem várias viaturas (em teoria de diversos tipos) e que empregam vários motoristas. A gestão que é feita destas empresas implica que após a criação da empresa se possam adicionar viaturas e motoristas, e que uma mesma viatura possa ser conduzida por motoristas diferentes (em tempos diferentes).

6 Viaturas com Fila de Espera

O UMeR possibilita que algumas viaturas possam ser definidas como possuindo um fila de espera, que possibilita que quando a viatura está indisponível os pedidos sejam adicionados a uma fila de

espera. Quando o veículo fica disponível são executadas todas as viagens inseridas na fila de espera, pela ordem de chegada. Os veículos com fila de espera possibilitam este comportamento, sendo que os demais não exibem este comportamento e nessa situação se a viatura estiver indisponível não será candidata a efectuar viagens.

7 Requisitos básicos

Identificam-se, de seguida, os requisitos básicos que o programa deverá cumprir: O programa deverá ainda fornecer uma interface de utilizador que permita acesso às funcionalidades.

- O estado da aplicação deverá estar pré-populado com um conjunto de dados significativos, que permita testar toda a aplicação no dia da entrega.
- Registar um utilizador, quer cliente quer motorista.
- Validar o acesso à aplicação utilizando as credenciais (email e password), por parte dos clientes e dos motoristas;
- 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 efectuadas (entre datas);
- ter acesso, no perfil de motorista, à listagem das viagens efectuadas (entre datas);
- indicar o total facturado 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 facturado;
- gravar o estado da aplicação em ficheiro, para que seja possível retomar mais tarde a execução.

8 Relatório

O relatório deve descrever o trabalho realizado para desenvolver a aplicação solicitada. No mínimo, devem ser abordados os seguintes pontos:

- Capa com identificação da Unidade Curricular e do grupo.
- Breve descrição do enunciado proposto.
- Descrição da arquitectura de classes utilizada (classes, atributos, etc.) e das decisões que foram tomadas na sua definição.

- Descrição da aplicação desenvolvida (ilustração das funcionalidades).
- Discussão sobre como seria possível incluir novos tipos de viaturas e motoristas na aplicação.

9 Salvaguarda do estado da aplicação

O programa deve permitir que em qualquer momento se possa guardar em ficheiro a informação existente em memória sobre os utilizadores, viaturas, viagens, etc. A gravação deve ser feita de forma a permitir que o estado que foi gravado seja recuperado novamente. Na altura da entrega do projecto deve ser também entregue um estado (guardado em ficheiro) que possa ser carregado durante a apresentação. Este estado deve conter dados significativos e que permitam testar toda a aplicação.

10 Patamares de classificação do projecto

Este projecto de POO tem previstos dois patamares de dificuldade em função dos requisitos anteriormente identificados:

1. Requisitos básicos de criação de viaturas, clientes, motoristas e registo de viagens: nota máxima 16 valores;
2. Itens anteriores mais gestão de factores de aleatoriedade, na duração da viagem e fiabilidade das viaturas, e criação de viaturas com fila de espera e empresas de táxis UMeR : nota máxima 20 valores

11 Cronograma

A entrega do projecto far-se-á de forma faseada, nas seguintes *milestones*:

1. Entrega de projecto BlueJ com uma versão inicial das declarações das classes (pelo menos com as variáveis de instância). Entrega da composição do grupo.
   Data Limite: 5 de Maio (esta fase é eliminatória, isto é, os grupos que não entregarem não poderão submeter o projecto final).
2. Entrega final de código e relatório de projecto (feita por via electrónica no elearning)
   Data Limite: 2 de Junho
3. Apresentação presencial do projecto
   Semana de: 5 a 9 de Junho