Paradigmas de Programação

2018/2019

Projeto (TP3)

Enunciado

O trabalho prático de Paradigmas de Programação (PPROG) será realizado em articulação com a unidade curricular de Engenharia de Software (ESOFT). O trabalho de PPROG consiste na implementação das funcionalidades definidas nos seguintes Casos de Uso do projeto que está a ser desenvolvido em ESOFT, intitulado "Sistema de Gestão de Prestação de Serviços Domésticos":

- Caso de Uso UC6 – Efetuar pedido de prestação de serviços (ator cliente)
- Caso de Uso UC9 – Indicar disponibilidade diária de prestação de serviços (ator prestador de serviços)(este caso de uso foi introduzido na iteração 2 de ESOFT)

Para além das funcionalidades relativas aos casos de uso UC6 e UC9, pretende-se ainda a implementação de uma outra funcionalidade: Atribuir pedido de prestação de serviço a um prestador de serviço (ator administrativo).
Esta funcionalidade deverá permitir atribuir um prestador de serviço a um serviço solicitado por um cliente, tendo em consideração o cumprimento dos seguintes critérios:

- A afetação de um prestador a um serviço deve ser feita de forma a minimizar a distância entre o local da prestação do serviço (definido pelo código postal do local de prestação do serviço) e uma das zonas geográficas de atuação do prestador de serviços (definidas também pelo código postal da zona geográfica);
- A afetação de um prestador a um serviço deverá respeitar a existência de compatibilidade entre o horário de disponibilidade do prestador e o horário indicado pelo cliente para a realização do serviço.

De forma a calcular a distância entre códigos postais, será usada a distância euclidiana entre o ponto central das zonas representadas por cada código postal. O ponto central de cada código postal é definido por coordenadas de latitude e longitude em graus, definidas num ficheiro de texto com o seguinte formato:

Código postal; Latitude (graus); Longitude (graus)
4100-000; 41.149304; -8.610880

A distância entre pontos definidos pelas suas coordenadas pode ser determinada usando o seguinte código:

```
double distancia (double lat1, double lon1, double lat2, double lon2) {
        // shortest distance over the earth's surface
        // https://www.movable-type.co.uk/scripts/latlong.html
        final double R = 6371e3;
        double theta1 = Math.toRadians(lat1);
        double theta2 = Math.toRadians(lat2);
        double deltaTheta = Math.toRadians(lat2 – lat1);
        double deltaLambda = Math.toRadians(lon2 – lon1);
        double a = Math.sin(deltaTheta / 2) * Math.sin(deltaTheta / 2)
```

```
                + Math.cos(theta1) * Math.cos(theta2)
                * Math.sin(deltaLambda / 2) * Math.sin(deltaLambda / 2);
        double c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
        double d = R * c; // distância em metros

        return d;
}
```

A implementação desta funcionalidade deve ter em conta a possibilidade de no futuro serem considerados critérios alternativos de afetação de prestadores de serviço.

Na implementação do projeto devem ser considerados os requisitos da iteração 2 do trabalho de ESOFT. Os enunciados das iterações 1 e 2 encontram-se em anexo. Os respetivos artefactos podem ser consultados no endereço https://bitbucket.org/pafomaio/esoft2018-2019-resolucao/ (os artefatos relativos à iteração 2 estarão disponíveis a partir do dia 15 de abril). Este repositório é partilhado pelo que o seu conteúdo não deve ser alterado.

No mesmo repositório está disponível o código de algumas classes Java que pode ser usado na implementação do trabalho de PPROG.

De forma a que a aplicação desenvolvida seja funcional, será necessário criar objetos de várias classes (cujo código está disponível). A implementação deverá incluir um mecanismo a executar no arranque da aplicação que permita criar os objetos das várias classes a partir de dados armazenados em ficheiros de texto. Por exemplo, o excerto de um ficheiro usado para armazenar informação acerca de instâncias da classe Categoria (de serviços) que possui os atributos "código" e "descrição" poderia ser:

cat1; limpeza
cat2; reparação de canalizações
...

Desta forma, após o arranque da aplicação, existirão em memória objetos contendo a descrição de categorias, serviços, áreas geográficas, clientes, endereços postais dos clientes e prestadores de serviço.

Os requisitos funcionais da implementação incluem:

- implementação das funcionalidades descritas;
- implementação de uma classe que instancie todas as classes necessárias para que as funcionalidades descritas possam ser utilizadas/testadas durante a apresentação do trabalho; esta classe deve criar instâncias de classes como por exemplo Categoria, Cliente, Servico, entre outras, já definidas no modelo de *design,* e ainda de outras classes a definir, como por exemplo PrestadorServico;
- implementação de um mecanismo que permita garantir a persistência dos dados, através do seu armazenamento num ficheiro binário, e posterior recuperação (implementação de mecanismos de leitura e de escrita); esta funcionalidade deverá aplicar-se apenas à classe PedidoPrestacaoServico;
- implementação de uma interface com o utilizador que permita usar as funcionalidades implementadas; a interface poderá ser baseada em

consola ou poderá consistir numa interface gráfica (esta última com valorização adicional – ver tópico avaliação);
- implementação de um mecanismo de instanciação de classes a partir do conteúdo de um ficheiro de texto – o ficheiro de texto deverá estar disponível no momento da apresentação do projeto contendo informação para a instanciação de todas as classes necessárias ao funcionamento da aplicação.

Os requisitos não funcionais incluem:
- desenvolvimento em linguagem Java e num projeto Maven;
- documentação do código, usando a ferramenta Javadoc;
- diagrama de classes em formato PDF;
- documento contendo a descrição das funcionalidades implementadas;
- implementação de testes unitários usando JUnit excetuando métodos getter, setter e toString;
- geração do relatório de cobertura de testes usando o plugin JaCoCo;
- utilização do repositório Bitbucket e da ferramenta de controlo de versões Git – **o repositório deverá ser privado e partilhado com o professor das aulas práticas – o nome do repositório deve respeitar o seguinte formato: PPROG_Turma_NúmeroAluno1_NúmeroAluno2_TP3**;
- **execução regular de commits por parte de cada um dos elementos do grupo durante todo o período de desenvolvimento do projeto**.

Normas de Funcionamento

Constituição dos grupos

O trabalho será realizado em grupos constituídos por dois alunos da mesma turma PL. Os alunos deverão informar por e-mail os respetivos professores acerca da constituição dos grupos até ao dia **18 de abril de 2019**. O não cumprimento deste requisito implicará a não avaliação do trabalho.
No caso de não ser possível proceder à formação de um grupo constituído por elementos da mesma turma PL, serão admissíveis situações excecionais de grupos constituídos por elementos de diferentes turmas, desde que devidamente justificadas e após aprovação por parte dos professores das aulas PL.

Avaliação

A avaliação levará em consideração fatores como a implementação de funcionalidades, utilização de boas práticas de codificação, qualidade da documentação e apresentação.
A interface com o utilizador terá um peso de 10% na avaliação do trabalho. No caso de ser implementada uma interface de consola, o respetivo item de avaliação será avaliado no máximo em 5%, sendo este máximo de 10% no caso de ser implementada uma interface gráfica.

Alunos que não frequentam ESOFT

Sugere-se que os alunos que não frequentem ESOFT constituam grupo com um aluno que se encontre a frequentar ESOFT. Serão disponibilizados no site da Wiki de ESOFT propostas de modelo de casos de uso, modelo de domínio e modelo de

design (diagrama de classes e diagrama de sequência).

Entrega
A entrega será realizada através de submissão no Moodle até às **23h30 do dia 18 de maio de 2019 (sábado)**. Esta entrega é constituída por um único ficheiro em formato ZIP, não sendo consideradas submissões de ficheiros com outros formatos. O ficheiro ZIP deve cumprir os seguintes requisitos:

- O nome do ficheiro deve seguir o seguinte formato **PPROG_Turma_NúmeroAluno1_NúmeroAluno2_TP3** (exemplo: PPROG_1DA_1870049_1120999_TP3.zip);
- O ficheiro ZIP deve incluir o seguinte:
    - Pasta contendo o projeto Maven, cujo nome segue as regras definidas para o nome do ficheiro ZIP;
    - Pasta com nome Doc contendo a documentação em formato HTML gerada pela ferramenta Javadoc;
    - Ficheiro em formato PDF com o nome requisitos.pdf contendo a enumeração das funcionalidades implementadas, assim como uma breve descrição de cada uma delas;
    - Ficheiro em formato PDF com o nome diagramaClasses.pdf contendo o diagrama de classes implementado.

Não serão consideradas para avaliação submissões que não cumpram completamente os requisitos definidos para a entrega. Poderão ser feitas várias submissões, sendo considerada apenas a última. As submissões devem ser efetuadas por ambos os elementos do grupo.

Apresentação
Após a submissão, os trabalhos terão que ser apresentados pelos membros do grupo ao respetivo professor das aulas práticas durante a última semana de aulas (12ª semana).

Nota importante
A utilização do repositório Bitbucket e da ferramenta de controlo de versões Git é obrigatória. A não existência deste repositório, assim como a falta de evidências acerca de *commits* regulares implica a não avaliação do trabalho.