Programação Modular
Documentação

Trabalho Prático 1 - Rede Social de Pesquisadores

Alunos

Andre Taiar Marinho Oliveira
Gustavo Henrique Alves Pereira

Professor
Douglas Macharet

Departamento de Ciência da Computação
UFMG - Universidade Federal de Minas Gerais
Belo Horizonte, Minas Gerais

Abril / 2015

Sumário

1 Introdução

Uma rede social pode ser definida como uma rede de interações sociais e relacionamentos pessoais [1]. Outra definição, mais popular, se refere a um website ou aplicação que permite que usuários se comuniquem. Apesar deste último conceito ser o mais presente quando se fala em redes sociais, o primeiro é útil para a compreensão de vários tipos de interações.

Este trabalho se refere a uma rede melhor descrita pela primeira definição. Supondo uma rede social fictícia de pesquisadores, relacionados por suas publicações, deseja-se obter métricas tais como a popularidade de dado pesquisador na rede ou a relevância de publicações.

Sendo um trabalho da disciplina de Programação Modular, o objetivo é criar um software de qualidade[2]. Para isto foi dada atenção aos fatores de qualidade externos, que são os percebidos pelo usuário, no caso o corretor do trabalho, e internos, relativos aos programadores. Imediatamente perceptível é a necessidade dos fatores internos, já que num trabalho consistindo de múltiplos programadores o entendimento mútuo é fundamental. Um pouco menos diretamente observável mas de fundamental importância são os fatores externos que, como mencionado, serão notados pelo corretor. Por exemplo, um código que violasse o fator 'correção' - isto é, não funcionasse conforme sua especificação - iria produzir resultados indesejados, o que obviamente é uma falha. Notando e aplicando as boas práticas de programação aprendidas esperam-se melhorias no fluxo de trabalho e resultados.

1.1 Visão geral

O programa recebe uma lista de parâmetros contendo os nomes dos arquivos necessários para obter os dados, como os pesquisadores, artigos e veículos. Após a leitura e processamento das informações os dados desejados são gravados em arquivos.

2 Implementação

A implementação do programa foi planejada a partir de uma perspectiva *bottom-up*: os itens componentes do problema foram examinados, implementados e então conectados uns com os outros.

2.1 Hierarquias

A primeira estrutura imaginada foi a hierarquia de Pesquisadores, por ser a de visualização e importância mais imediatamente aparente. Um `Pesquisador` é ente que realiza pesquisa, escrevendo artigos. Ele possui um número identificador único, que o diferencia de outros. O Pesquisador possui uma maneira de calcular a sua popularidade, calculada como o seu peso (métrica baseada na autoria de artigos) somado ao número de artigos e citações. Ele foi modelado como uma classe abstrata, pois cada pesquisador será, de fato, um Graduado, Mestre ou Doutor, mas também se deseja que Graduados, Mestres e Doutores compartilhem de certos atributos em comum. Desta maneira decidiu-se criar a hierarquia que começa em Pesquisador e termina em Doutor, passando por Graduado e Mestre. A necessidade de compartilhamento de atributos foi o que fez com que a decisão do nível mais alto ser uma classe abstrata e não uma interface fosse tomada. Para que se mantivessem as características entre cada nível a visibilidade dos métodos e atributos comuns foi estabelecida como `protected`.

Seguindo a hierarquia de Pesquisador, o `Graduado` é um pesquisador que já se formou. Ele é um Pesquisador que, além das informações comuns, possui horas trabalhadas na iniciação científica e no estágio de docência. Ele também calcula sua popularidade como sendo a popularidade do Pesquisador mais as horas de iniciação científica e do estágio de docência. Do ponto de vista da orientação a objetos, este é um bom exemplo do reuso de códigos. Seria possível escrever, para cada tipo de pesquisador, um método diferente para o cálculo de popularidade. Percebeu-se, porém, que o cálculo se mantinha na hierarquia, de modo que cada nível mais especializado realizava este cálculo como sendo o do nível superior mais algum item próprio. Este também foi um item responsável pela decisão de se utilizar uma classe abstrata em detrimento de uma interface: o Pesquisador, ainda que não possa ser instanciado, possui cálculo de popularidade. Como interfaces não permitem implementação de métodos, é mais conveniente a criação de uma classe abstrata.

Na mesma hierarquia o `Mestre` é mais especializado que `Graduado`, além de poder orientar alunos. Seu cálculo de popularidade inclui o cálculo dos níveis superiores mais um valor baseado no número de alunos que orienta.

Por fim, o `Doutor` é o mais especializado, sendo um `Pesquisador` que possui características de `Mestre` e `Graduado`, além de poder orientar qualquer tipo de pesquisador.

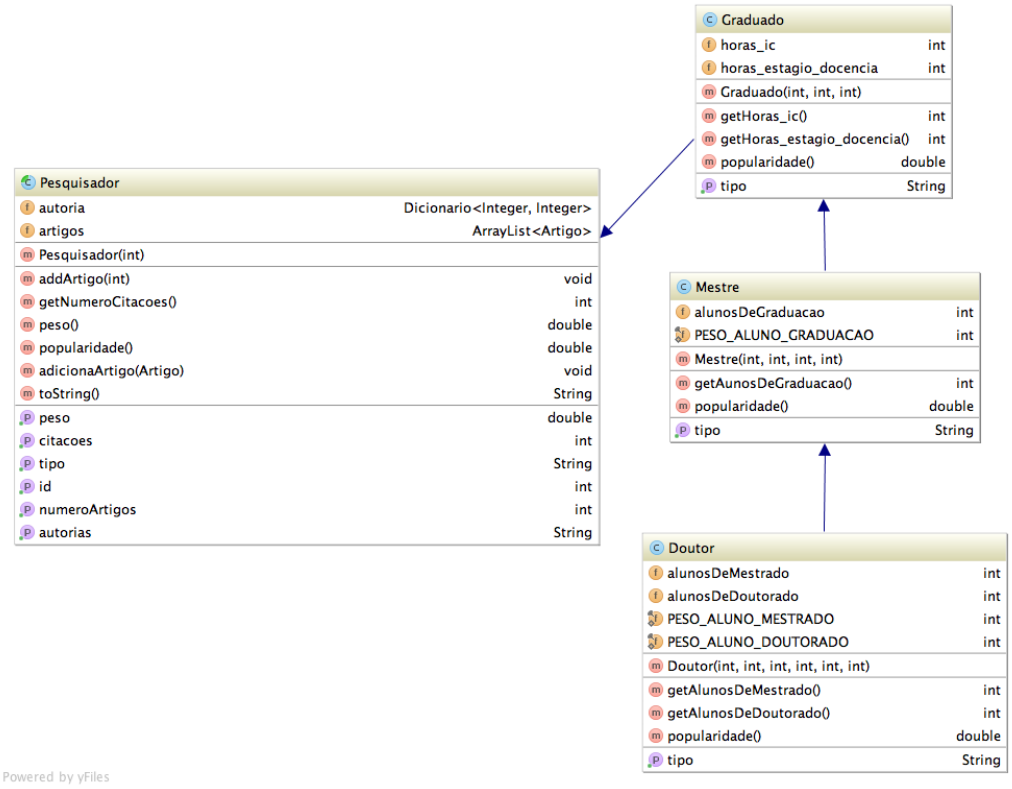

Figura 1: Hierarquia do pacote `Pesquisadores`

A hierarquia de Pesquisador foi colocada num pacote [1], denominado `Pesquisadores`.

O passo seguinte foi a definição da hierarquia de Veículos de Publicação. Um veículo de publicação é um agrupamento de artigos. De maneira análoga ao que foi feito para a classe `Pesquisador`, criou-se uma classe abstrata `VeiculoDePublicacao` contendo as características comuns a todos os veículos.

Os dois tipos de veículo de publicação são `Conferencia` e `Revista`. Os dois são herdeiros diretos de `VeiculoDePublicacao`, estando no mesmo nível. Ambos contém coleções de artigos e se diferenciam na maneira de calcular o fator de impacto - que, aliás, é semelhante para a superclasse abstrata até certo ponto, justificando a implementação referida.

[1] Namespace que organiza um conjunto de classes e interfaces relacionadas[5]

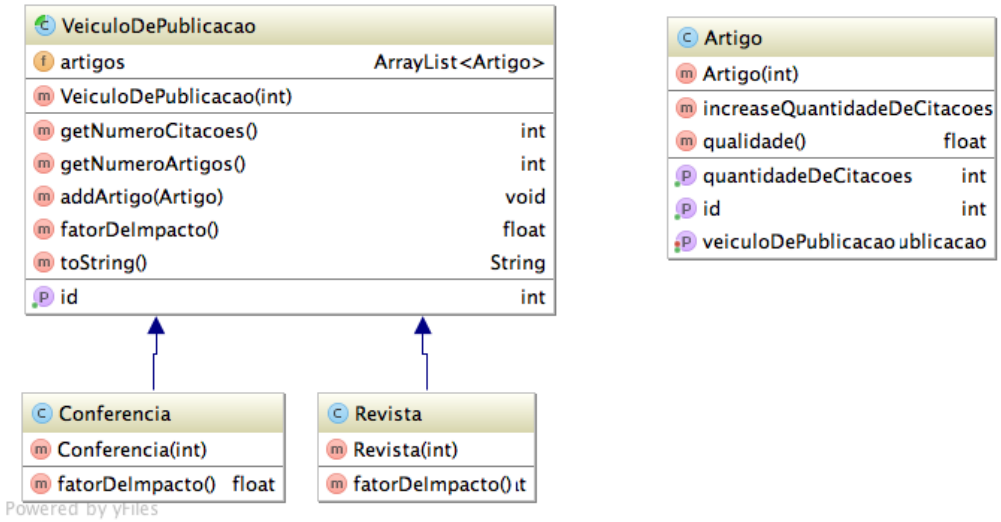

Figura 2: Estutura do pacote de `VeiculosDePublicacao`

Os artigos são representados na classe `Artigo`. Esta classe pode ser considerada a base de todos os relacionamentos no programa. Um pesquisador tem seu peso base calculado pelas autorias em artigos, além de se relacionar com outros pesquisadores por autorias em conjunto. Um artigo também se relaciona com outros artigos, podendo citar e ser citado. Esta classe foi objeto de algumas das decisões mais interessantes do projeto. Um artigo possui um número identificador único e o número de vezes em que foi citado. A classe faz uso de composição ao armazenar uma referência ao veículo de publicação ao qual pertence, o que é útil no relacionamento entre o veículo e o artigo. Há também o cálculo da qualidade do artigo, dependente do veículo ao qual pertence e a quantidade de citações.

`VeiculoDePublicacao`, `Revista`, `Conferencia` e `Artigo` estão presentes no pacote `VeiculosDePublicacao`.

Os pacotes anteriores completam as estruturas que representam o problema.

2.2 Utilitários

Para que o programa pudesse ser criado e funcionasse corretamente foram desenvolvidos outros tipos de estrutura além dos componentes originais do

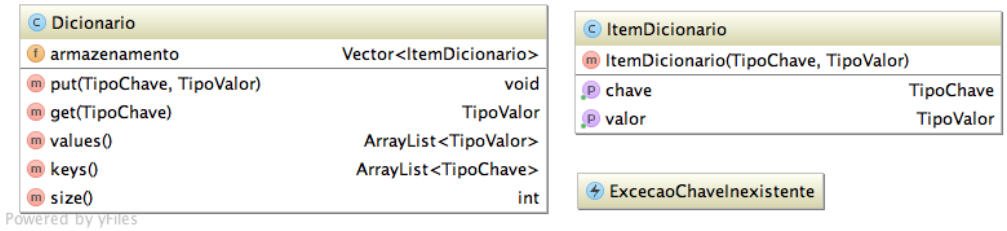

Figura 3: Estutura do pacote `Utilitarios`

problema, que foram reunidos no pacote `Utilitarios` por conveniência.

Durante o desenvolvimento surgiu a ideia do uso da estrutura `HashMap` do Java, que é um container de dados parametrizados indexado por um objeto também parametrizado, útil para os propósitos de seleção dos arquivos de entrada e armazenamento de pesquisadores. Após uma parte do código funcionando com o uso de `HashMap` surgiu a informação de que a estrutura não deveria ser utilizada. Neste ponto seria por demais custoso trocar a implementação por outra ideia, então optamos por desenvolver uma estrutura com as mesmas operações disponíveis. Internamente, o `Dicionario` que substitui o `HashMap` é simplesmente um `Vector` de outra classe chamada `ItemDicionario`, também presente no pacote de utilitários. Este item do dicionário contém atributos respectivos a chave e ao valor respectivos, ambos parametrizados. Quando um elemento está para ser inserido no dicionário verifica-se a existência da mesma chave. Caso a chave não exista ainda, o par chave e valor é adicionado. Caso a chave exista, o valor respectivo é modificado. Verificar se a chave existe é um processo de busca linear. É conhecida a estrutura tabela hash, que calcula uma chave para cada valor e pode ser implementada com um acesso direto a vetor, mas acreditamos que o método utilizado é suficiente para este caso. Obviamente, num caso em que a quantidade de dados seja imensa, uma implementação de tabela hash seria melhor. O pacote também contém a exceção `ExcecaoChaveInexistente`, lançada quando é realizada a tentativa de acesso a uma chave inexistente é tentada.

2.3 Comunicação com o mundo externo

Por fim, o pacote `EntradaESaida` contem classes responsáveis pela entrada e saída de dados. Toda a lógica envolvendo a leitura dos arquivos e carrega-

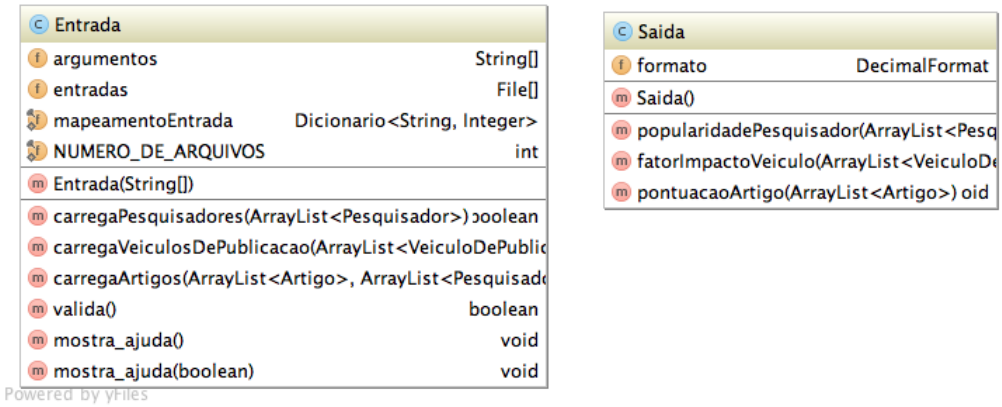

Figura 4: Estutura do pacote de `EntradaESaida`

mento dos dados é chamada neles.
É importante notar que os módulos de entrada e saída são responsáveis apenas pela inserção e exportação dos dados. Todos os cálculos são realizados pelas entidades específicas desginadas. Assim, quando se executa um método que imprime num arquivo a popularidade dos pesquisadores, o cálculo é realizado de acordo com o método que obtém a popularidade do pesquisador, determinado no contrato daquela hierarquia. O módulo de saída apenas executa a chamada, não tendo qualquer informação sobre como a popularidade é obtida por cada pesquisador.

Em `Entrada` estão os métodos de carregamento dos itens. Eles são responsáveis pelo carregamento de pesquisadores, artigos e veículos de publicação. Alguns métodos acessórios estão presentes, para validar entrada e informar ao usuário sobre problemas ocorridos. Nestes métodos encontramos a maioria dos tratamentos de exceção do programa, geralmente relacionados a arquivos não encontrados e tentativa de acesso a chaves não existentes no dicionário.

`Saida` contém métodos que obtém os dados já calculados e os imprime num arquivo, para consulta. As exceções lançadas neste módulo se referem principalmente a erros de IO.

2.4 Funcionamento

O programa possui seu ponto de chamada na classe `Tp1_pm`. Esta classe contém estruturas de armazenamento para pesquisadores, veículos e artigos,

além de instâncias de módulos de entrada e saída. O módulo de entrada processa os arquivos de texto passados como parâmetro, carregando nas estruturas de armazenamento os dados dos arquivos. Após esta fase, o módulo de saída salva as informações especificadas no trabalho (popularidade dos pesquisadores, fator de impacto dos veículos e pontuação dos artigos), que são obtidas dos dados armazenados.

3 Testes

Foram disponibilizados dois conjuntos de testes para que o programa pudesse ser verificado. Eles foram caracterizados pelo número de Pesquisadores presentes no arquivo "pesquisadores.txt", sendo assim os conjuntos "5 pesquisadores"e "15 pesquisadores". Seguem as saídas, visto que se assume as entradas como conhecidas pelo corretor.

3.1 5 pesquisadores

`popularidade_pesquisador.txt`

```
1;731.0000
2;35.5000
3;64.0000
4;341.5000
5;509.5000
```

`fatorImpacto_veiculo.txt`

```
1;1.4286
2;1.0000
3;1.3000
```

pontuacao_artigo.txt

```
1;0.0000
2;0.0000
3;0.0000
4;0.0000
5;0.0000
6;0.0000
7;1.4286
8;0.0000
9;0.0000
10;0.0000
11;1.3000
12;0.0000
13;0.0000
14;0.0000
15;0.0000
16;0.0000
17;0.0000
18;0.0000
19;0.0000
20;2.8571
21;0.0000
22;0.0000
23;1.3000
24;1.3000
25;0.0000
26;0.0000
```

3.2 15 pesquisadores

popularidade_pesquisador.txt

```
1;1356.6667
2;264.7500
3;357.8333
4;355.8333
5;639.2500
6;522.2500
7;778.8333
8;732.0000
9;377.5000
10;357.1667
11;405.0000
12;420.6667
13;595.4167
14;305.0000
15;366.0000
```

fatorImpacto_veiculo.txt

```
1;4.0161
2;4.2308
3;4.0862
4;3.9444
```

pontuacao_artigo.txt

1;15.7778 2;12.0484 3;11.8333 4;4.0862 5;8.1724
6;12.0484 7;24.5172 8;8.4615 9;23.6667 10;8.0323
11;4.2308 12;12.2586 13;7.8889 14;21.1538 15;16.3448
16;12.2586 17;24.0968 18;8.1724 19;7.8889 20;16.9231
21;20.4310 22;12.0484 23;12.0484 24;19.7222 25;7.8889
26;16.3448 27;8.1724 28;25.3846 29;8.0323 30;12.2586
31;12.0484 32;12.2586 33;3.9444 34;16.3448 35;0.0000
36;12.0484 37;8.0323 38;15.7778 39;4.2308 40;0.0000
41;21.1538 42;24.0968 43;11.8333 44;12.2586 45;16.9231
46;20.4310 47;29.6154 48;4.0161 49;8.4615 50;11.8333
51;8.4615 52;27.6111 53;8.4615 54;12.6923 55;12.6923
56;11.8333 57;20.4310 58;11.8333 59;15.7778 60;8.0323
61;12.2586 62;12.2586 63;11.8333 64;12.0484 65;12.0484
66;8.4615 67;21.1538 68;16.3448 69;16.3448 70;16.9231
71;12.0484 72;3.9444 73;21.1538 74;28.6034 75;16.9231
76;8.0323 77;12.0484 78;16.3448 79;8.4615 80;11.8333
81;25.3846 82;16.0645 83;16.9231 84;29.6154 85;12.2586
86;35.5000 87;28.6034 88;8.0323 89;4.2308 90;8.1724
91;7.8889 92;4.2308 93;25.3846 94;7.8889 95;8.0323
96;4.0862 97;8.0323 98;20.4310 99;7.8889 100;7.8889
101;0.0000 102;4.0161 103;8.4615 104;20.0806 105;8.1724
106;7.8889 107;0.0000 108;0.0000 109;16.0645 110;11.8333
111;21.1538 112;0.0000 113;0.0000 114;16.9231 115;7.8889
116;11.8333 117;11.8333 118;16.9231 119;12.6923 120;24.0968
121;20.0806 122;4.0862 123;16.9231 124;8.4615 125;12.0484
126;4.0862 127;16.0645 128;4.0862 129;20.0806 130;12.0484
131;21.1538 132;16.0645 133;4.0161 134;28.1129 135;20.0806
136;4.0862 137;11.8333 138;16.0645 139;16.0645 140;20.0806
141;8.0323 142;16.0645 143;7.8889 144;16.9231 145;15.7778
146;20.4310 147;12.2586 148;8.1724 149;7.8889 150;7.8889
151;4.0161 152;8.0323 153;4.0862 154;4.0161 155;8.1724
156;12.6923 157;19.7222 158;16.3448 159;12.0484 160;8.0323
161;8.4615 162;16.3448 163;3.9444 164;4.0862 165;16.3448
166;0.0000 167;12.6923 168;12.0484 169;16.3448 170;20.4310
171;12.6923 172;11.8333 173;16.0645 174;4.2308 175;12.2586
176;12.2586 177;8.0323 178;8.4615 179;8.0323 180;20.0806
181;11.8333 182;8.0323 183;8.0323 184;7.8889 185;24.5172
186;12.2586 187;15.7778 188;7.8889 189;0.0000 190;12.0484
191;19.7222 192;4.2308 193;3.9444 194;21.1538 195;12.6923
196;3.9444 197;8.1724 198;11.8333 199;11.8333 200;15.7778
201;7.8889 202;28.6034 203;16.0645 204;11.8333 205;24.0968
206;16.9231 207;4.2308 208;7.8889 209;27.6111 210;4.2308
211;20.4310 212;8.4615 213;4.0862 214;12.0484 215;4.0862
216;16.3448 217;3.9444 218;16.9231 219;8.1724 220;16.9231
221;8.4615 222;12.0484 223;12.2586 224;8.0323 225;8.0323
226;19.7222

4 Conclusão

Este primeiro trabalho prático da disciplina de Programação Modular envolveu a implementação de uma rede social de pesquisadores. Além da expansão do conceito e métricas interessantes, a proposta de fato do trabalho era o uso das boas práticas de programação e tecnologia de orientação a objetos ensinados nas aulas.

A aplicação dos conhecimentos obtidos foi realmente alcançada. Especialmente por se tratar de um trabalho em dupla, foram travadas várias discussões quanto aos tipos de dados, visibilidade, decisões de implementação e modularização.

Um dos casos mais interessantes e problemáticos foi o já citado impedimento do uso de `HashMap`, que nos levou a buscar uma solução rápida para que o código não sofresse modificações muito severas num estágio já avançado do desenvolvimento. A implementação ocorreu relativamente livre de problemas, e se mostrou estável. Ela foi primeiro testada como um projeto separado e só após a verificação da qualidade foi inserida no trabalho final.

Outra particularidade devida ao fator "dupla"foi a utilidade da criação de módulos. Com todas as partes do código tendo suas responsabilidades bem definidas era simples decidir em qual ponto um dos membros do grupo trabalharia em dado momento para implementar uma funcionalidade ou corrigir um erro, sem fazer alterações drásticas que pedissem uma reestruturação vasta do projeto. O benefício da definição dos contratos logo no início foi de imenso valor.

Por fim, acreditamos ter alcançado os objetivos propostos pelo trabalho. Nos familiarizamos com aspectos da orientação a objetos aplicados com o uso da linguagem Java. Conseguimos criar um código confiável, correto e modularizado.

5 Referências

[1] Oxford Dictionary, http://www.oxforddictionaries.com/us/definition/american_english/social-network

[2] MEYER, Bertrand. *Object Oriented Software Construction*, 2 ed. Prentice Hall, 1997.

[3] Oracle Docs: Class HashMap. http://docs.oracle.com/javase/7/docs/api/java/util/HashMap.html

[4] Oracle Docs: Class ArrayList. http://docs.oracle.com/javase/7/docs/api/java/util/ArrayList.html

[5] Oracle The Java Tutorials: What is a Package? https://docs.oracle.com/javase/tutorial/java/concepts/package.html