

Algoritmos e Estruturas de Dados

Gestão de uma Biblioteca

Turma 4 - Grupo B

José Peixoto	200603103	ei12134@fe.up.pt
Paulo Faria	201204965	ei12135@fe.up.pt
Pedro Moura	201306843	up201306843@fe.up.pt

7 de Novembro de 2014

Conteúdo

1	Descrição da solução implementada	3
2	Lista de casos de utilização da aplicação	3
3	Relato das principais dificuldades encontradas no desenvolvimento do trabalho	4
4	Indicação do esforço dedicado por cada elemento do grupo	4
4.1	Notas em grupo	4
4.1.1	José Peixoto	4
4.1.2	Paulo Faria	4
4.1.3	Pedro Moura	4
A	UML	5

Resumo

Foi-nos proposto o desenvolvimento de uma aplicação em C++ para gestão de uma biblioteca que abrangesse livros, leitores, funcionários e supervisores. A biblioteca teria necessariamente que permitir, de forma controlada, o empréstimo de livros a leitores e gestão de ambos pelos funcionários. Os empréstimos, quando possíveis, teriam um período máximo de duração de 7 dias para cada livro, podendo o leitor incorrer numa multa caso não o devolvesse dentro deste prazo. Os supervisores estariam individualmente a cargo de uma equipa composta por funcionários.

1 Descrição da solução implementada

Após um período de interpretação do guião que nos foi fornecido, determinámos, através de esquemas, a relação entre as classes necessárias para implementar o pedido recorrendo a UML. Decidimos que a classe *Library* teria um papel basilar ao armazenar e manipular as estruturas de dados essenciais à gestão de uma biblioteca, nomeadamente contentores de livros, leitores e funcionários. Estabelecemos uma relação de hierarquia de classes entre pessoas, leitores e funcionários, sendo estes dois últimos subclasses da classe abstrata *Person*. Considerámos inapropriado a criação de uma subclasse *Supervisor* em relação à classe *Employee*, por se tratar de uma solução quase forçada que recorre a uma relação hierárquica entre classes que, afinal, não diferem significativamente. Como um supervisor é sempre à partida um empregado, permitimos a sua promoção, recorrendo a um booleano *supervisor*, membro dado da classe *Employee*.

Na conceptualização de empréstimos, foi necessária a classe *Borrow* que associaria a cada empréstimo um livro, um leitor e um empregado responsável pela operação. Ainda a propósito da criação e gestão correta de um empréstimo, sentimos a necessidade de recorrer a uma classe que nos permitisse representar a data de forma suficientemente flexível para determinar facilmente as diferenças entre duas datas (empréstimo e entrega do livro).

2 Lista de casos de utilização da aplicação

A aplicação permite a criação, edição e remoção de livros, leitores, funcionários e empréstimos. É possível a visualização simplificada e ordenação de livros, leitores e funcionários. Na gestão de empréstimos, a aplicação também admite o cálculo de multa, caso o leitor não devolva o livro dentro do prazo estipulado.

3 Relato das principais dificuldades encontradas no desenvolvimento do trabalho

Mesmo tendo investido algum tempo no planeamento inicial, encontrámos algumas dificuldades na definição de uma representação ideal dos conceitos pedidos e que permitisse de igual forma satisfazer as operações requeridas. Fomos, portanto, de forma iterativa, reformulando as nossas abordagens a cada problema/incompatibilidades que foram surgindo. Tivemos também que transpor vários problemas no uso do software de gestão de repositórios *Git*, por não termos conhecimentos que nos permitissem fluxos de trabalho paralelos, sem conflitos, aquando da junção do código.

4 Indicação do esforço dedicado por cada elemento do grupo

4.1 Notas em grupo

Tentámos desde o início, trabalhando em grupo, fazer o esboço das classes e separação de tarefas de forma equitativa. Fomos trabalhando ora por turnos, ora por conferência via *Skype*.

4.1.1 José Peixoto

Tal como os restantes membros do grupo, participei desde o início na esquematização de uma solução do que nos fora pedido. Depois das tarefas divididas, estive mais envolvido nas classes *Library*, *Reader*, *Algorithms e Interface*, colaborando também, sempre que necessário, nas classes a cargo dos meus colegas. Além disso assumi, voluntariamente, a concretização do relatório final, incluindo as autoavaliações redigidas pelos restantes colegas.

4.1.2 Paulo Faria

Colaborei sobretudo nas classes *Interface*, *Book*, *Library*, mas tal como os meus colegas, sempre que o era necessário, também dei uma ajuda em outras classes.

4.1.3 Pedro Moura

Estive mais envolvido nas classes *data*, *exception*, *borrow*, *persons* e na leitura e escrita de ficheiros, colaborando também em outras partes do programa, quando necessário.

