Algoritmos e Estruturas de Dados

# Gestão de uma Biblioteca

**Segunda Parte**

Ângela Cardoso, Catarina Terra e Maria Miranda


4 de Janeiro de 2015

# Conteúdo

**1 Introdução** **2**

**2 Descrição da Solução Implementada** **3**

**3 Diagrama de Classes UML** **6**

**4 Casos de Utilização** **7**

**5 Principais Dificuldades** **10**

**6 Distribuição de Trabalho Pelos Elementos do Grupo** **11**

**7 Conclusão** **12**

# Capítulo 1

# Introdução

No âmbito da disciplina de Algoritmos e Estruturas de Dados, foi-nos proposto implementar em C++ uma aplicação que permita a gestão de uma biblioteca. Entretanto, como segunda parte desse trabalho, alteramos e acrescentamos novas estruturas, de forma a permitir a manutenção de pedidos de empréstimo, de leitores inativos e de livros disponíveis.

O sistema inicial continha informações sobre livros, leitores, funcionários e empréstimos de livros. A aplicação permitia registar e gerir os empréstimos efetuados na biblioteca, assim como toda a informação já referida. Entretanto, adicionaram-se três novas estruturas à biblioteca: uma árvore binária de pesquisa com os livros disponíveis para emprestar; uma tabela de dispersão com os leitores inativos, que não efetuam empréstimos há mais de um ano; uma fila de prioridade com os pedidos de empréstimo dos livros indisponíveis.

Mantiveram-se as restrições anteriores. Cada leitor não pode ter mais de 3 livros emprestados em simultâneo. Por casa dia de atraso o leitor incorre numa multa de 0,25€ por dia na primeira semana e de 0,50€ por dia nas semanas seguintes. Um livro pode ser emprestado por um período máximo de 1 semana ou não pode ser emprestado. Cada empréstimo é feito por um funcionário. Os funcionários podem ou não ser supervisores e cada supervisor é responsável por um ou mais funcionários, nunca por outro supervisor, devendo esta distribuição ser equilibrada.

Os livros passam a ter informação sobre o ano de edição, estando os livros disponíveis ordenados na árvore por ano de edição, título e autores.

Os pedidos estão ordenados por data e, em caso de serem do mesmo dia, por prioridade dos leitores, estudantes primeiro, depois crianças e finalmente adultos.

Ao longo deste documento descreve-se a aplicação que desenvolvemos para gestão de um biblioteca, com foco nos desenvolvimentos da segunda parte do trabalho.

# Capítulo 2

# Descrição da Solução Implementada

Para conter toda a informação de uma biblioteca, implementou-se a classe **Biblioteca**, que é constituída pelas seguintes estruturas:

- *livros* - vetor com todos os livros da biblioteca;
- *funcionarios* - vetor com todos os funcionários, incluindo os supervisores e o administrador;
- *leitores* - vetor com todos os leitores da biblioteca;
- *emprestimos* - vetor com todos os empréstimos da biblioteca;
- *pedidos* - vetor com todos os pedidos de empréstimo da biblioteca;
- *disponiveis* - árvore binária de pesquisa com os livros disponíveis para empréstimo;
- *inativos* - tabela de dispersão com os leitores inativos há mais de um ano;
- *utilizadores* - vetor com todos os utilizadores do sistema informático da biblioteca.

Com exceção dos utilizadores, quando um objeto é removido (por exemplo, no caso de devolução de um empréstimo), não desaparece, apenas é alterado para indicar que já não existe, sendo-lhe acrescentada a data de remoção. Assim, para a classe **Livro** (respectivamente, **Funcionario**, **Leitor**, **Emprestimo**, **Pedido**) existe a subclasse **Livro_old** (respectivamente, **Funcionario_old**, **Leitor_old**, **Emprestimo_old**, **Pedido_old**), para onde é enviado um livro (respectivamente, funcionário, leitor, empréstimo) quando é removido.

As classes **Livro**, **Funcionario**, **Leitor**, **Emprestimo**, **Pedido** e **Utilizador** são subclasses da classe **Object**, que tem como único atributo o código de identificação **ID**.

O primeiro passo na utilização da aplicação é efetuar o login. Para isso é necessário indicar o ID e a password, sendo então determinado o tipo de acesso do utilizador, 0 se for

administrador (gerente da biblioteca), 1 se for um supervisor e 2 se for um funcionário. ID, password e acesso são atributos da classe **Utilizador**.

Uma das funções mais procuradas numa biblioteca é o empréstimo de livros a leitores. A classe **Emprestimo** faz a gestão dessas funções e tem como parâmetros um apontador para o livro do emprestimo, um apontador para o funcionario que fez o emprestimo, um apontador para o leitor do emprestimo, a data do emprestimo e um contador de emprestimos feitos na biblioteca (atuais e antigos).

Não estando nenhum exemplar do livro disponível para empréstimo, é possível deixar um pedido de empréstimo. A classe **Pedido** faz a gestão dessas funções e tem como parâmetros um apontador para o livro do pedido, um apontador para o funcionario que fez o pedido, um apontador para o leitor do pedido, a data do pedido e um contador de pedidos feitos na biblioteca (atuais e antigos). Além de serem mantidos num vetor na biblioteca, os pedidos são mantidos em filas de prioridade dos respetivos livros, uma vez que quando são de livros distintos, não há qualquer relação de prioridade entre dois pedidos.

Tal como as classes **Emprestimo** e **Pedido**, as classes **Livro**, **Funcionario** e **Leitor**, também possuem contadores do numero dos seus objetos na biblioteca. Estes contadores são utilizados para permitir a atribuição automática de códigos de identificação a cada um destes objetos. Assim, se adicionamos um novo livro à biblioteca, por exemplo, e em toda a sua existência (atuais e antigos) a biblioteca teve 13 livros, o ID do novo livro será 14.

A classe **Livro** tem como atributos o ano_edicao, o titulo, um vetor com os nomes dos autores, o tema, o ISBN, a cota do livro na biblioteca, o num_paginas, a edicao, o número de exemplares desse livro, o número ex_disponiveis de exemplares que estão disponíveis para emprestar e um vetor emp_livro com apontadores para os empréstimos de cada exemplar do livro. Sendo assim, cada exemplar corresponde a um índice do vetor de empréstimos e, se o exemplar estiver disponível, no respetivo índice do vetor está um apontador nulo.

Além dos atributos já referidos, o único atributo da classe **Funcionario** é o nome do funcionário. A classe **Supervisor** é subclasse de **Funcionario** e tem como atributo adicional um vetor funcionarios_sup de apontadores para os funcionários supervisionados pelo supervisor. Uma vez que o número de funcionários supervisionados por cada supervisor

deve ser equilibrado, a distribuição dos funcionários pelos supervisores é feita de forma automática, garantindo o equilíbrio, de cada vez que:

- é adicionado um funcionário;
- é removido um funcionário ou supervisor;
- é promovido um funcionário a supervisor;
- é despromovido um supervisor a funcionário.

Existe ainda a classe **Administrador**, como subclasse de **Funcionario** (sem atributos adicionais). Apenas é adicionado um administrador à biblioteca, com o ID 0 e o nome Administrador. Este age como gerente da biblioteca, sendo o único que tem acesso total e que pode adicionar, remover, promover e despromover funcionários. É também o único que pode efetuar a manutenção dos utilizadores.

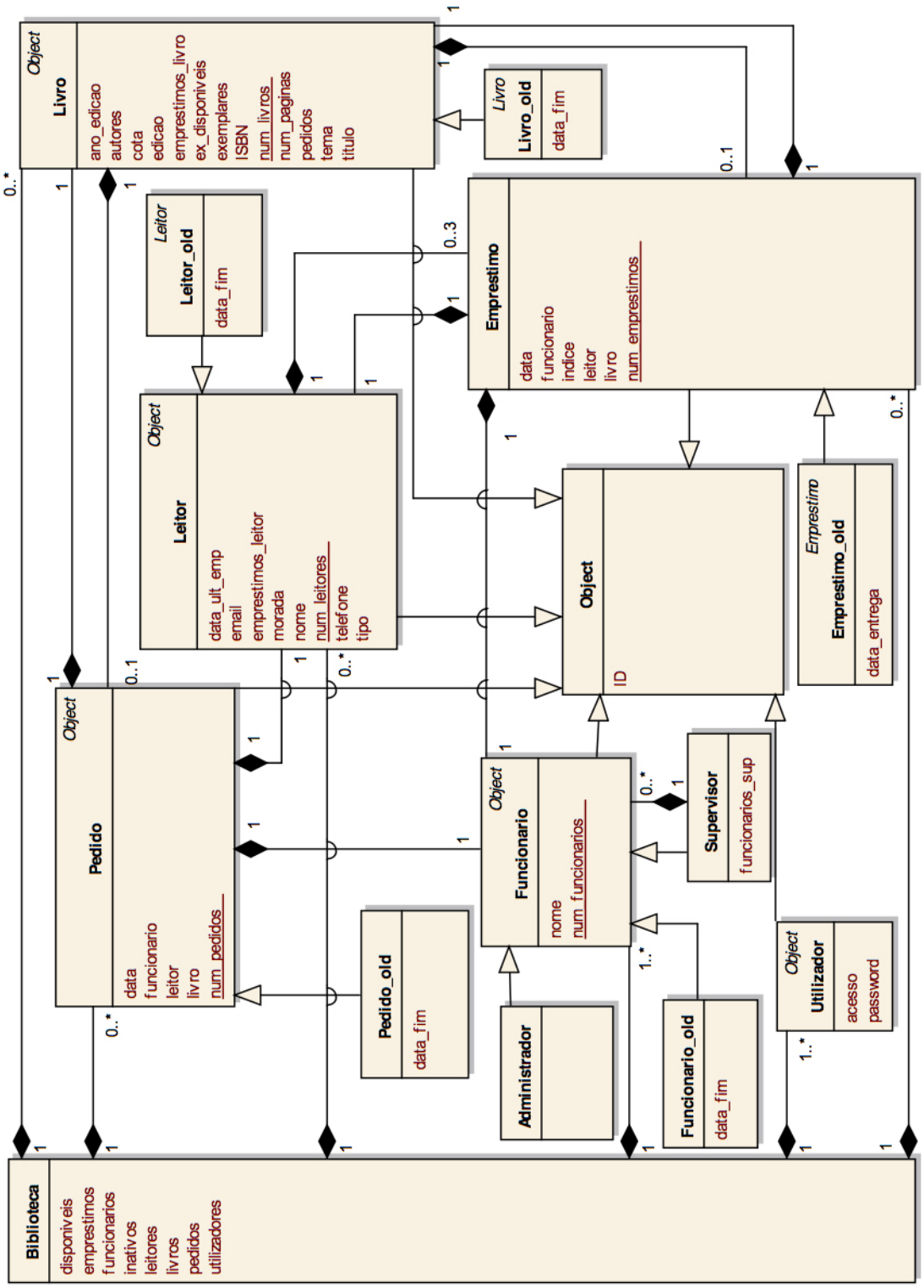
Além dos mencionados acima, os atributos da classe **Leitor** são o nome, o tipo (estudante, criança ou adulto), o telefone, o email e um vetor emprestimos_leitor com apontadores para os seus empréstimos.

A interação do utilizador com a aplicação é feita primariamente através de uma série de menus na consola, de seleção numérica. Uma sequência de opções nestes menus, conduz a uma função da classe **Biblioteca**, que por sua vez poderá chamar uma ou mais funções das restantes classes mencionadas. A gestão destes menus é feita na classe **Menu**, subclasse de **Biblioteca**, que como atributo adicional tem um **Utilizador_online**, consoante o qual é determinado o tipo de acesso e as funções disponíveis.

Além das classes mencionadas, existem ainda várias classes onde estão parametrizadas as exceções a usar quando não existe um determinado objeto, tentamos adicionar um empréstimo a um leitor com 3 empréstimos, tentamos emprestar um livro já emprestado, tentamos pedir um livro disponível, tentamos remover um objeto ocupado (livro emprestado, por exemplo), tentamos utilizar um ficheiro que não existe, etc.

# Capítulo 3

# Diagrama de Classes UML



# Capítulo 4

# Casos de Utilização

A aplicação para gestão de uma biblioteca construída permite registar informaticamente uma série de atividades relacionadas com o funcionamento normal de uma biblioteca.

1. Consultas
    1.1. Livros
    1.2. Empréstimos
    1.3. Pedidos
    1.4. Leitores
    1.5. Funcionários
    1.6. Supervisores
    1.7. Utilizadores
2. Empréstimos
    2.1. Adicionar
    2.2. Devolver
    2.3. Consultar atrasos
    2.4. Consultar atrasos por leitor
    2.5. Consultar livros atrasados
    2.6. Consultar antigos
3. Pedidos
    3.1. Adicionar
    3.2. Desistir

3.3. Consultar antigos

4. Livros

4.1. Consultar disponíveis

4.2. Consultar emprestados

4.3. Consultar por tema

4.4. Consultar antigos

4.5. Adicionar

4.6. Remover

4.7. Alterar

4.7.1. Ano de edição

4.7.2. Título

4.7.3. Autores

4.7.4. Tema

4.7.5. ISBN

4.7.6. Cota

4.7.7. Número de páginas

4.7.8. Edição

4.7.9. Tudo

5. Leitores

5.1. Adicionar

5.2. Remover

5.3. Alterar

5.3.1. Nome

5.3.2. Tipo

5.3.3. Telefone

5.3.4. Email

5.3.5. ISBN

        5.3.6. Morada

        5.3.7. Tudo

    5.4. Consultar inativos

    5.5. Consultar antigos

6. Funcionários

    6.1. Adicionar

    6.2. Remover

    6.3. Promover

    6.4. Despromover

    6.5. Consultar antigos

7. Utilizadores

    7.1. Adicionar

    7.2. Remover

A disponibilidade destas funções depende do nível de acesso do utilizador. De forma geral, um supervisor pode, além de todas as tarefas acessíveis a um funcionário, adicionar e remover livros e consultar informação sobre funcionários. O administrador é o único que pode fazer a manutenção dos funcionários e dos utilizadores.

# Capítulo 5

# Principais Dificuldades

Nesta segunda parte, a maior dificuldade de implementação foi fazer as alterações necessárias ao que já tinha sido feito anteriormente. Além do que se acrescentou, foi preciso fazer várias mudanças às classes **Livro** e **Leitor**. Cada alteração teve diversos impactos nas outras classes e nas várias funções. Foi necessário alterar tudo isso e os ficheiros de texto, garantindo que a aplicação continuava a funcionar.

Além disso, surgiram algumas dificuldades relacionadas com o facto de se guardarem objetos na árvore de pesquisa, nas filas de prioridade e na tabela de dispersão, em vez de apontadores como se faz nos vetores. Foi necessário colocar objetos na árvore e nas filas, para que estas respeitassem as respetivas ordens dos elementos. No entanto, isto implicou a adição de passos quando se altera um livro ou um leitor, para garantir que os respetivos elementos da árvore e da tabela de dispersão são alterados também.

De resto, uma vez que todas as ferramentas já são conhecidas e o domínio da própria linguagem também é maior, foi mais simples implementar a segunda parte do trabalho. Entre outras coisas, tivemos oportunidade de melhorar vários pontos daquilo que foi feito anteriormente.

# Capítulo 6

# Distribuição de Trabalho Pelos Elementos do Grupo

Em relação à primeira parte do trabalho, houve aqui melhorias significativas, nomeadamente a nível de comunicação entre os elementos do grupo.

Embora de forma muito menos acentuada, a disponibilidade dos elementos do grupo para o trabalho continuou a não ser a mesma. Desta forma, a Ângela acabou por realizar a maior parte das tarefas de codificação. A Catarina e a Maria, além de alguma contribuição na codificação e na alteração dos ficheiros de texto, dedicaram-se a validar o funcionamento da aplicação, tendo elaborado e efetuado uma bateria de testes por forma a detectar possíveis erros de codificação.

# Capítulo 7

# Conclusão

Pela forma como o trabalho se desenvolveu, além do acumular de conhecimento, cremos que a experiência e o resultado final foram mais positivos nesta segunda parte.

O trabalho manteve-se muito interessante e foi gratificante poder aplicar os conhecimentos adquiridos na parte final da disciplina. Mais uma vez ajudou-nos a sedimentar o conhecimento dos diferentes tipos de estrutura utilizados.

Como sempre, se o tempo o permitisse, certamente seriam várias as alterações e melhorias introduzidas. Mas de uma forma geral estamos satisfeitas com o resultado final.