Trabalho Prático - Fase 1

Programação Orientada a Objetos

Docente: Luís Gonzaga Martins Ferreira

Aluno: Rodrigo Lopes Ferreira Nº 31525

Repositório GitHub:

https://github.com/LFtech6/OOP_Project

Índice

Índice de Figuras *3*

Resumo *4*

1 - Introdução *5*

1.1 - Motivação **5**

1.2 - Enquadramento **5**

1.3 - Objetivos **5**

1.4 - Metodologia de Trabalho **6**

1.5 - Plano de Trabalho **6**

1.6 - Estrutura do Documento **6**

2 - Estado da Arte, Enquadramento Teórico e Prático *7*

2.1 - Programação Orientada a Objetos (POO) **7**

2.1.1 – Encapsulamento 7

2.1.2 – Abstração 8

2.1.3 – Herança 8

2.1.4 – Polimorfismo 8

2.2 – Modelação de Sistemas de Gestão de Condomínios **8**

2.3 – Enumerações e Tipificação de Dados **9**

2.3.1 – TipoDocumento 9

2.3.2 – MetodoPagamento 9

2.4 – Estruturas de Dados Utilizadas **9**

2.5 – Relação com Sistemas de Gestão Reais **9**

3 – Trabalho Desenvolvido *10*

3.1 – Análise do Domínio **10**

3.2 – Especificação das Classes **11**

3.3 – Implementação da Estrutura Base **12**

3.3.1 – Classe Abstrata Pessoa 12

3.3.2 – Classe Condomino 12

3.3.3 – Classe Fraccao 12

3.3.4 – Classe Condominio 13

3.3.5 – Classes Financeiras 13

3.3.6 – Classes Administrativas 13

3.3.7 – Classe Permilagem 13

3.4 – Documentação XML e Organização do Código **13**

3.5 – Síntese da Fase 1 ..... **14**

4 – Resultados e Discussão 15

5 – Conclusão 16

6 - Glossário e Siglas 16

7 - Referências 17

Índice de Figuras

Figura 1 - Diagrama UML 14

Resumo

Este relatório descreve o trabalho realizado na Fase 1 do projeto de Programação Orientada a Objetos. O objetivo principal consistiu na identificação e implementação da estrutura base de um sistema para gestão de condomínios, recorrendo aos princípios fundamentais da POO.

Nesta fase, foram definidas as classes centrais do domínio, os seus atributos, propriedades, construtores e métodos essenciais (sem implementação lógica). Aplicaram-se conceitos como encapsulamento, herança, abstração, modularidade e enumerações, garantindo uma base sólida para as fases seguintes.

1 - Introdução

O presente capítulo pretende introduzir o problema abordado neste trabalho, o seu contexto académico, os objetivos estabelecidos, a metodologia de desenvolvimento, o plano de trabalho seguido, e ainda a estrutura do documento.

1.1 - Motivação

A gestão de condomínios exige organização rigorosa de informações relacionadas com proprietários, frações, documentos, pagamentos, reuniões e despesas. A ausência de um sistema estruturado origina inconsistências e dificuldades de controlo administrativo. Este trabalho surge como resposta a essa necessidade de organização, aplicando os conhecimentos adquiridos em Programação Orientada a Objetos.

1.2 - Enquadramento

O projeto foi desenvolvido no âmbito da unidade curricular de Programação Orientada a Objetos, do 2.º ano da Licenciatura em Engenharia de Sistemas Informáticos (LESI – IPCA). A Fase 1 tem como objetivo estabelecer a arquitetura base de classes conforme as instruções dadas pelo docente.

1.3 - Objetivos

Os principais objetivos da Fase 1 foram:

- Identificar as entidades fundamentais do sistema.
- Criar a classe abstrata base (Pessoa).
- Definir atributos e propriedades das classes.
- Implementar construtores adequados.
- Criar métodos base (retorno 0), a serem desenvolvidos em fases posteriores.
- Utilizar enumerações para valores categorizados.
- Organizar o código segundo boas práticas de POO e do repositório fornecido.

1.4 - Metodologia de Trabalho

A metodologia seguida incluiu:

1. Análise do domínio e definição das entidades.
2. Estruturação das classes com base nos pilares da POO.
3. Desenvolvimento incremental das classes e enumerações.
4. Integração da documentação XML para cada classe.
5. Revisão da consistência entre atributos, propriedades e construtores.

1.5 - Plano de Trabalho

O trabalho foi desenvolvido em 2 fases, tendo esta primeira quatro etapas:

- Análise conceptual do domínio.
- Levantamento e criação das classes necessárias.
- Implementação de atributos, propriedades e métodos placeholder.
- Documentação do código e revisão geral.

1.6 - Estrutura do Documento

O presente relatório está organizado da seguinte forma:

- **Resumo** – Apresenta uma visão global do trabalho, contextualizando a problemática da gestão de condomínios e os objetivos definidos para a Fase 1.
- **Capítulo 1 – Introdução** – Introduz o tema e o contexto do projeto, as motivações, os objetivos, a metodologia adotada, o plano de trabalho e a atual estrutura do documento.
- **Capítulo 2 – Estado da Arte** – Apresenta os fundamentos teóricos e práticos relevantes para o projeto, com destaque para os princípios da Programação Orientada a Objetos e para a modelação de sistemas de gestão.
- **Capítulo 3 – Trabalho Desenvolvido** – Descreve as etapas práticas da Fase 1, incluindo a análise do domínio, a identificação das classes, a especificação dos seus atributos e relações, bem como a implementação da estrutura de código em C#.

- **Capítulo 4 – Resultados** – Apresenta e discute os resultados obtidos nesta fase, analisando a consistência da arquitetura definida, as decisões de modelação tomadas e a preparação do sistema para fases futuras.
- **Capítulo 5 – Conclusão** – Resume os principais pontos do trabalho, os objetivos alcançados, as dificuldades encontradas e possibilidades de evolução e melhoria do sistema.
- **Capítulo 6 – Glossário** – Reúne os principais termos técnicos e siglas utilizados ao longo do documento, facilitando a compreensão do leitor.
- **Referências** – Lista as fontes bibliográficas e materiais de apoio utilizados no desenvolvimento do trabalho, de acordo com o regulamento em vigor.

2 - Estado da Arte, Enquadramento Teórico e Prático

A gestão de condomínios envolve um conjunto de processos administrativos, financeiros e documentais que requerem organização rigorosa. A complexidade deste tipo de sistemas justifica a utilização de metodologias de desenvolvimento estruturado, assentes nos princípios da Programação Orientada a Objetos (POO). Neste capítulo apresenta-se o enquadramento teórico e os conceitos que sustentam a modelação do sistema desenvolvido na Fase 1.

2.1 - Programação Orientada a Objetos (POO)

A Programação Orientada a Objetos é um paradigma de programação que organiza o software através da definição de classes e objetos. Cada classe representa um conceito do mundo real, contendo dados (atributos) e comportamentos (métodos).

Este paradigma destaca-se pela promoção da modularidade, manutenção simplificada e elevada reutilização de código. Os pilares fundamentais aplicados neste projeto são:

2.1.1 – Encapsulamento

O encapsulamento consiste na proteção dos atributos internos de uma classe, garantindo que o acesso aos mesmos é feito através de propriedades controladas.
Neste projeto, todos os atributos são privados e expostos através de propriedades públicas ***(get/set)***, assegurando o controlo sobre os dados.

2.1.2 – Abstração

A abstração permite representar conceitos gerais através de classes abstratas.
A classe Pessoa é um exemplo disso: define atributos e comportamentos comuns a todos os tipos de utilizadores, mas não pode ser instanciada diretamente. Esta classe serve de base (superclasse) à classe *Condomino*.

2.1.3 – Herança

A herança permite que uma classe derive de outra, reutilizando código e comportamentos comuns.
A classe *Condomino* herda da classe Pessoa, beneficiando da estrutura base e adicionando atributos específicos.

2.1.4 – Polimorfismo

Este princípio permite que diferentes classes implementem métodos com o mesmo nome mas comportamentos distintos.
Na Fase 1, o polimorfismo é demonstrado através do método abstrato IdentificarPessoa(), que será implementado de forma concreta nas subclasses.

2.2 – Modelação de Sistemas de Gestão de Condomínios

Os sistemas de gestão de condomínios devem abranger um conjunto de entidades e relações que representam a estrutura administrativa habitual.
Entre os elementos mais comuns encontram-se:

- Condóminos (proprietários das frações)
- Frações do edifício
- Condomínio como entidade agregadora
- Reuniões e atas associadas
- Pagamentos e quotas
- Despesas do condomínio
- Documentos administrativos

A modelação realizada na Fase 1 obedece a esta estrutura, representando cada uma das entidades como uma classe independente, promovendo modularidade e clareza arquitetural.

2.3 – Enumerações e Tipificação de Dados

As enumerações desempenham um papel importante ao definir conjuntos de valores fechados, garantindo maior segurança e consistência no código.
Foram criadas duas enumerações:

2.3.1 – TipoDocumento

Define categorias documentais como ata, contrato, relatório, comunicação, entre outros.
Esta enumeração evita o uso de valores arbitrários e facilita a interpretação e validação futura.

2.3.2 – MetodoPagamento

Indica os métodos disponíveis para pagamentos: transferência, MBWay, numerário, cheque ou outro.
A sua utilização impede erros comuns como valores inválidos, aumentando a robustez e clareza do sistema.

2.4 – Estruturas de Dados Utilizadas

Tendo em conta o âmbito da Fase 1, foram utilizadas estruturas simples, nomeadamente:

- Arrays para armazenar coleções de objetos (Condomino[])
- Tipos primitivos (string, double, int)
- Classes compostas como propriedades (ex.: um Condomino possui uma Fraccao)

Não foram utilizadas listas dinâmicas ou coleções genéricas, por não serem ainda necessárias nesta fase do projeto.

2.5 – Relação com Sistemas de Gestão Reais

A estrutura definida aproxima-se da forma como empresas reais de gestão de condomínios operam.
As tarefas administrativas implicam:

- Identificação de proprietários e respetivas frações
- Gestão de comunicações formais
- Cálculo proporcional de valores através da permilagem
- Registo de despesas e pagamentos efetuados
- Emissão e arquivo de documentos
- Planeamento de reuniões e organização de atas

A modelação realizada suporta estas necessidades e cria uma fundação adequada para as fases posteriores, onde a lógica e o comportamento real do sistema serão implementados.

3 – Trabalho Desenvolvido

Este capítulo descreve detalhadamente o trabalho realizado na Fase 1 do projeto, desde a análise inicial até à implementação das classes em C#. A abordagem seguida foi orientada para a construção de uma base arquitetural sólida, modular e facilmente extensível, de acordo com os princípios de Programação Orientada a Objetos e com o estilo de trabalho sugerido no repositório fornecido pelo docente.

3.1 – Análise do Domínio

A análise iniciou-se com a identificação das entidades fundamentais envolvidas no processo de gestão de condomínios.
Foram consideradas as necessidades comuns deste tipo de sistemas: registo de condóminos, caracterização das frações, cálculos proporcionais (permilagem), controlo financeiro (quotas, pagamentos e despesas), organização administrativa (reuniões) e documentação oficial (atas, contratos, relatórios, etc.).

A partir desta análise foram identificadas as seguintes entidades principais:

- Pessoa (conceito abstrato)
- Condómino
- Fração

- Condomínio
- Documento
- Reunião
- Despesa
- Quota
- Pagamento
- Permilagem

A existência de comportamentos comuns entre diferentes tipos de indivíduos levou à criação da classe abstrata Pessoa.
A classe Condomino surge como uma especialização desta, refletindo a realidade de que um proprietário é sempre uma pessoa, mas possui informação adicional relevante para o contexto condominial.

A separação entre entidades financeiras (Quota, Pagamento, Despesa) e administrativas (Documento, Reuniao) permitiu organizar o sistema de forma clara e modular.

3.2 – Especificação das Classes

Após a identificação das entidades, foi realizada a definição dos atributos, propriedades e relações entre classes. Este processo seguiu três critérios principais:

1. **Relevância real para o domínio**
   Cada atributo foi escolhido com base em necessidades reais de gestão condominial (ex.: morada de notificação, valor da fração, ordem de trabalhos).

2. **Coerência com POO**
   Todos os atributos foram mantidos como privados, sendo expostos através de propriedades públicas (*get/set*), respeitando o encapsulamento.

3. **Preparação para expansão futura**
   Alguns métodos foram implementados como placeholders (retornando 0) para serem desenvolvidos nas fases seguintes, sem comprometer a estrutura.

As relações definidas incluem, por exemplo:

- **Um Condómino tem uma Fração**

- **Um Condomínio tem vários Condóminos**
- **Uma Permilagem está associada a uma Fração**
- **Um Pagamento corresponde a uma Quota**
- **Um Documento pertence ao contexto do condomínio**

Enumerações foram igualmente definidas para garantir consistência:

- TipoDocumento
- MetodoPagamento

3.3 – Implementação da Estrutura Base

A implementação da estrutura base foi realizada em C#, respeitando o formato de organização ensinado pelo docente e aplicado no repositório de apoio.

3.3.1 – Classe Abstrata Pessoa

Define atributos e propriedades comuns (nome, nif, contacto), bem como um método abstrato (IdentificarPessoa()) que força as subclasses a implementar o seu próprio comportamento.

3.3.2 – Classe Condomino

Herda de Pessoa e contém atributos adicionais:

- Morada de notificação
- Fração associada
- Piso
- Permilagem

Foram implementados construtores completos e métodos simples.

3.3.3 – Classe Fraccao

Representa cada unidade física do condomínio, armazenando identificação, área e proprietário.

3.3.4 – Classe Condominio

Agrega informação global de um edifício: nome, localização, data de construção e o conjunto de condóminos.

3.3.5 – Classes Financeiras

- **Quota**: representa o valor mensal a pagar.
- **Pagamento**: regista a operação financeira, usando MetodoPagamento.
- **Despesa**: regista gastos do condomínio.

Cada uma possui propriedades adequadas e métodos placeholder (RegistarQuota(), RegistarPagamento(), etc.).

3.3.6 – Classes Administrativas

- **Documento**: representa atas, contratos, relatórios, entre outros.
- **Reuniao**: armazena a data, local e ordem de trabalhos.

Usam enums e seguem o mesmo padrão organizacional.

3.3.7 – Classe Permilagem

Guarda os valores base para cálculo da permilagem de uma fração, permitindo implementar posteriormente o respetivo método de cálculo.

3.4 – Documentação XML e Organização do Código

Todas as classes incluem documentação XML simples e clara:

- <summary> para descrição geral
- Comentários adicionais nos métodos e propriedades-chave
- Estrutura dividida em regiões (#region) para melhorar a legibilidade

Esta uniformidade facilita:

- consulta rápida,

- leitura estruturada,
- futura manutenção e expansão do sistema.

3.5 – Síntese da Fase 1

A Fase 1 concluiu com a:

- definição da arquitetura essencial do sistema;
- implementação de todas as classes com atributos e propriedades;
- documentação e organização do código;
- preparação lógica para etapas posteriores (e.g., cálculo real, validações, coleções dinâmicas).

Esta base serve como alicerce para a Fase 2, onde serão implementados comportamentos e relações dinâmicas entre entidades.

Segue aqui um anexo do diagrama UML que representa as minhas classes:

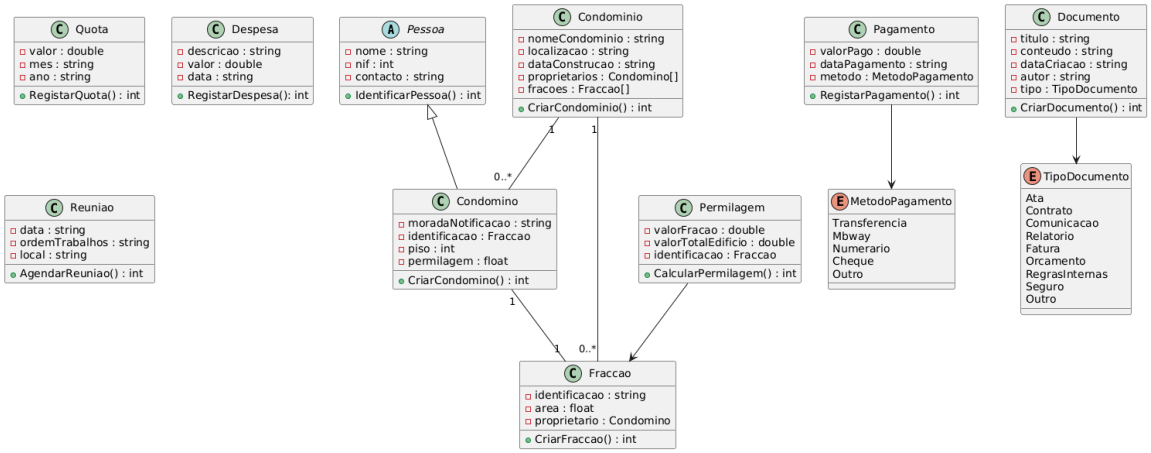

Figura 1 - Diagrama UML

4 – Resultados e Discussão

O resultado final desta fase é uma arquitetura de classes consistente, organizada e fácil de expandir para a Fase 2.
Foram respeitados todos os requisitos exigidos, garantindo:

- Coerência entre classes
- Correta utilização de herança
- Estrutura modular
- Preparação adequada para adicionar lógica e relacionamentos mais complexos

A estrutura obtida constitui uma fundação sólida para a implementação das funcionalidades reais do sistema.

5 – Conclusão

A Fase 1 permitiu estabelecer a estrutura fundamental para um sistema de gestão de condomínios. As classes implementadas representam adequadamente as entidades principais do domínio e encontram-se preparadas para receber a lógica de negócio em fases seguintes.

Foram consolidados conhecimentos de programação orientada a objetos e aplicados de forma consistente, nomeadamente herança, encapsulamento, abstração e modularidade.

6 - Glossário e Siglas

A seguir apresenta-se um glossário com os principais termos e siglas utilizados ao longo do relatório, por ordem alfabética.

- **POO** – Programação Orientada a Objetos
- ***Enum*** – Tipo enumerado para valores pré-definidos
- **Fração** – Unidade pertencente a um edifício em propriedade horizontal
- **Permilagem** – Valor proporcional usado para calcular contribuições num condomínio
- **Ata** – Registo oficial de reunião

7 - Referências

StackOverflow. (2025). Website para ver excertos de código de outras pessoas. Disponível em: **https://stackoverflow.com/**

W3Schools. (2025). Website utilizado para estudos sobre programação no geral. Disponível em:

https://www.w3schools.com/cs/

***GitHub*. (2025).** Repositório com relatório de inspiração. Disponível em:

https://github.com/LFtech6/EDA_TP

***GitHub*. (2025).** Repositório das aulas. Disponível em:

https://github.com/luferIPCA/LESI-POO-2025-2026

ChatGPT. (2025).

https://chatgpt.com