Ciência da Computação – Projeto de Software (2025/1)

TRABALHO PRÁTICO

Alunos: Thiago Henrique, Joao Pedro, Joao Paulo

1. Descrição do Padrão

O padrão de projeto Abstract Factory é um dos padrões criacionais do Catálogo GoF (Gang of Four). Seu principal objetivo é fornecer uma interface para criar famílias de objetos relacionados ou dependentes sem especificar suas classes concretas.

Esse padrão é muito útil em situações onde os produtos precisam ser utilizados em conjunto e devem ser compatíveis entre si, como é o caso de sistemas multiplataforma, montadoras de produtos com variações, entre outros.

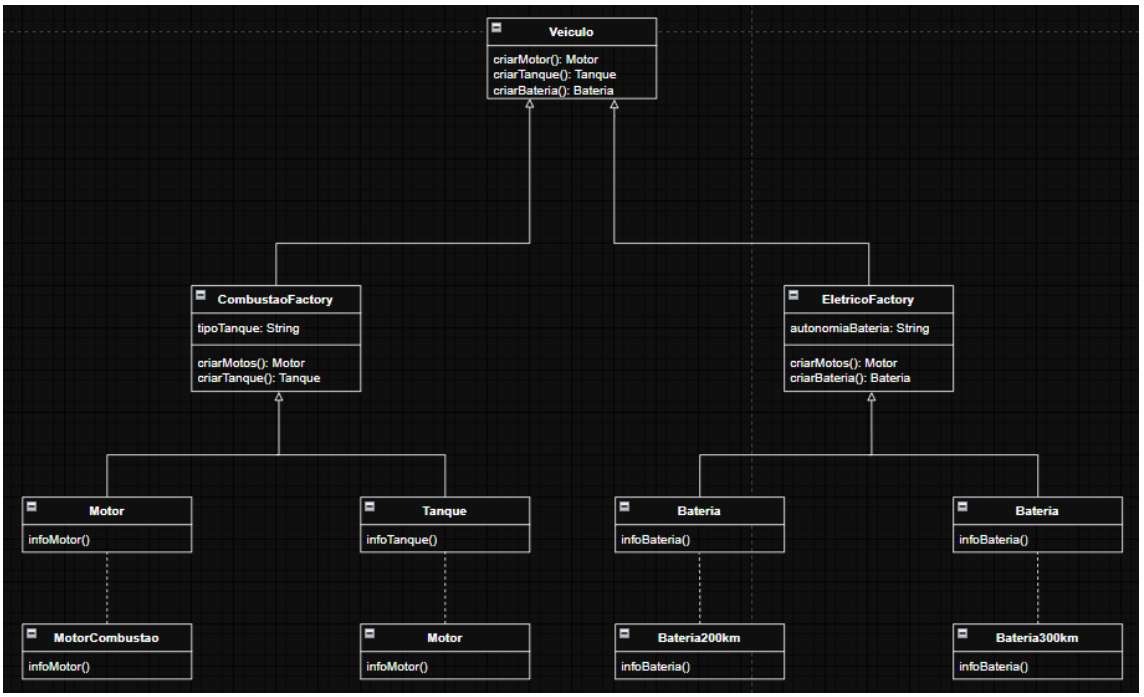
Problema que resolve: Evita o acoplamento direto entre o código cliente e as classes concretas de objetos que ele utiliza. Isso facilita a manutenção, evolução e expansão do sistema, promovendo maior flexibilidade.

2. Implementação do Padrão

A implementação do padrão Abstract Factory envolve os seguintes elementos principais:

- **Uma interface abstrata para a fábrica (`VeiculoFactory`);**
- **Múltiplas fábricas concretas (`CombustaoFactory` e `EletricoFactory`), cada uma criando um conjunto de produtos compatíveis entre si;**
- **Interfaces para os produtos: `Motor`, `Tanque` e `Bateria`;**
- **Implementações concretas dos produtos como `MotorCombustao`, `TanqueGasolina`, `Bateria300km`, etc.;**
- **Uma classe cliente (`Veiculo`) que recebe a fábrica como dependência e utiliza os objetos criados.**

Veja abaixo o diagrama de classes gerado para essa implementação:



Explicação do Diagrama:

- `VeiculoFactory` ***é a interface principal da fábrica;***
- `CombustaoFactory` ***e*** `EletricoFactory` ***implementam a lógica para criação dos componentes de acordo com o tipo de veículo;***
- ***Cada componente (motor, tanque e bateria) tem uma interface e suas implementações concretas;***
- ***A classe*** `Veiculo` ***usa a fábrica recebida para montar os componentes.***

3. Cenário de Aplicação do Padrão

Contexto: Uma montadora de veículos deseja criar um sistema de montagem em que o usuário possa montar um carro de acordo com suas preferências:

- ***Veículo a Combustão:***
  - ***Tipo de combustível: Gasolina ou Álcool***
- ***Veículo Elétrico:***
  - ***Autonomia desejada: 200 km ou 300 km***

Como o padrão foi aplicado: O sistema aplica o Abstract Factory para criar componentes compatíveis com o tipo de veículo escolhido. Isso evita erros de combinação (ex: motor elétrico com tanque de gasolina), promove a expansibilidade (poderemos facilmente adicionar outros combustíveis ou baterias), e desacopla a lógica de montagem do conhecimento sobre as classes concretas.

4. Diagrama de Classes e Explicação

O diagrama mostra:

- **A dependência da classe `Veiculo` com a interface `VeiculoFactory`;**
- **As implementações concretas para os tipos de motor, tanque e bateria;**
- **A separação clara entre abstração e implementação, o que demonstra o uso adequado do padrão.**

5. Implementação em Java

A solução foi desenvolvida em Java e permite que o usuário interaja via console para montar um veículo com as opções desejadas. O sistema é modular e permite futuras expansões com novos tipos de motores, combustíveis ou baterias.

6. Conclusão

O padrão Abstract Factory mostrou-se ideal para esse cenário de montagem de veículos, garantindo coesão, flexibilidade e manutenção facilitada do código. A solução entregue está funcional, extensível e cumpre os requisitos propostos na atividade.