

AULA 9 – PADRÃO *ABSTRACT FACTORY*

GS1020 – Programação Orientada a Objetos II

Prof. Dr. Murillo G. Carneiro
mgcarneiro@ufu.br



Material baseado nos slides cedidos pelo Prof. Rafael D. Araújo (FACOM/UFU)

Objetivo da aula

- Entender o funcionamento do padrão de projeto *Abstract Factory* (Fábrica Abstrata).

Fábrica Abstrata

- É um padrão de projeto de propósito **de criação** com escopo de **objetos**.
- Propõe uma solução para a criação de famílias de objetos sem expor as suas classes.
- Em outras palavras, é uma fábrica de fábricas.

Intenção

- Fornecer uma interface para criar famílias de objetos relacionados ou dependentes sem especificar suas classes concretas.

Quando usar

- Quando um sistema deve ser independente de como seus produtos são criados, compostos ou representados;
- Quando um sistema deve ser configurado como um produto de uma família de múltiplos produtos;
- Uma família de objetos-produto for projetada para ser usada em conjunto e você necessita garantir esta restrição;
- Você quer fornecer uma biblioteca de classes de produtos e quer revelar somente suas interfaces, não suas implementações.

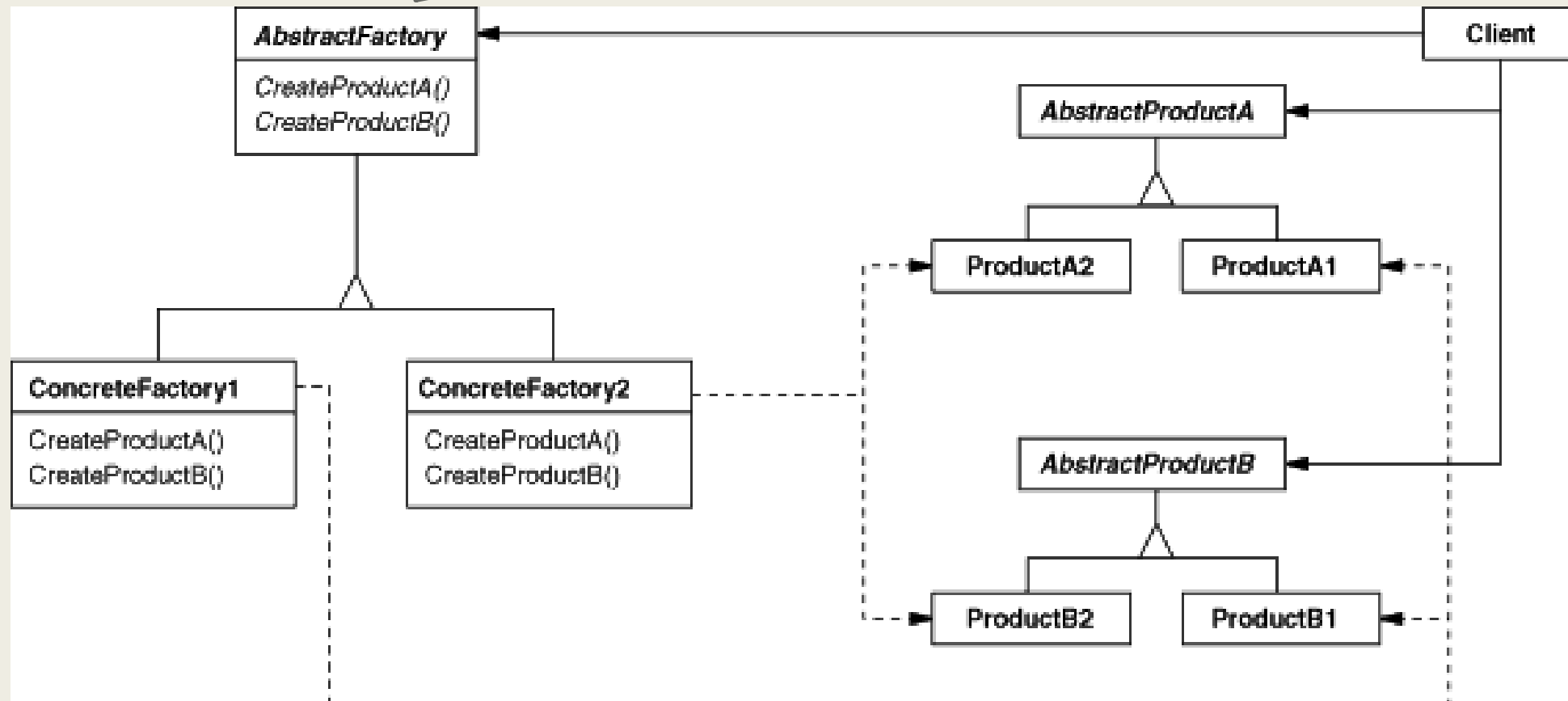
Elementos participantes

- **AbstractFactory**, declara uma interface para operações que criam objetos-produto abstratos.
- **ConcreteFactory**, implementa as operações que criam objetos-produto concretos.
- **AbstractProduct**, declara uma interface para um tipo de objeto-produto.
- **ConcreteProduct**, implementa a interface `AbstractProduct` (abstração de um produto concreto).
- **Client**, aplicação que utiliza a fábrica de família de objetos (conhece somente as interfaces/tipos abstratos).



Estrutura

A fábrica abstrata pode possuir diversas implementações concretas, cada uma capaz de fabricar instâncias de diferentes famílias de produtos concretos.

O cliente só conhece os tipos abstratos!



Benefícios

- Isola a criação de objetos de seu uso e cria famílias de objetos relacionados sem ter que depender de suas classes concretas desacoplamento
 - *Permite introduzir novos tipos derivados sem qualquer alteração ao código que usa a classe base*
- Permite trocar implementações concretas sem alterar o código que estas usam, mesmo em tempo de execução flexibilidade

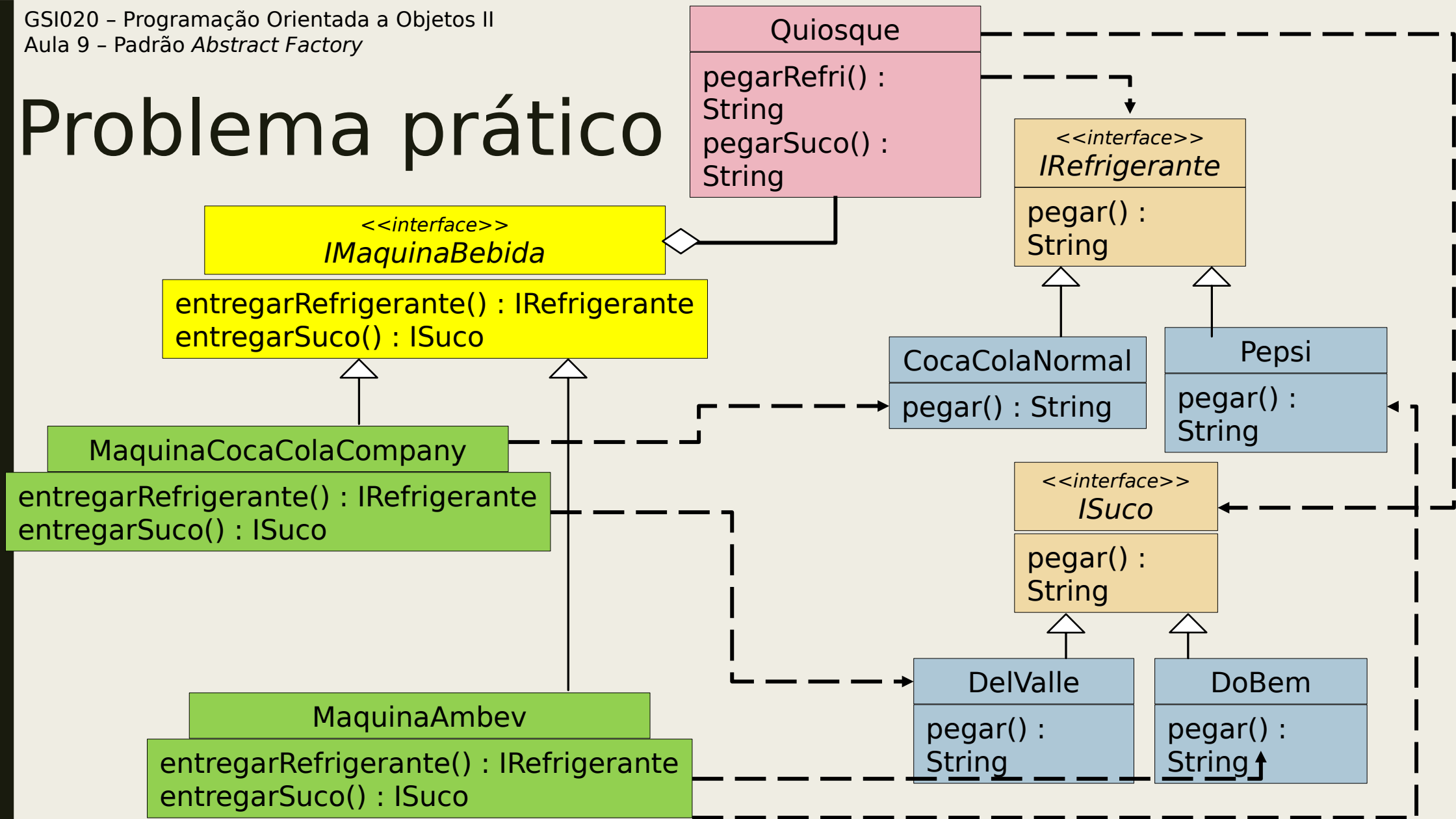
Desvantagens

- É difícil incluir novos tipos de produtos
 - *a FactoryAbstrata fixa o conjunto de produtos que podem ser criados. Por isso, criar um novo tipo de produto exige estender a interface da fábrica (o que leva a mudar a classe AbstractFactory e todas as suas subclasses)*

Problema prático

1) Implemente o cenário das máquinas de bebidas utilizando o padrão Fábrica Abstrata.

Problema prático



AbstractFactory AbstractProduct Client
ConcreteFactory ConcreteProduct

Problema prático

- 2) O método construtor da classe Quiosque deve receber uma fábrica abstrata (IMaquinaBebida).
- 3) No método main, crie um quiosque Coca-Cola.
 - *Imprima o resultado do método pegarRefri().*
 - *Imprima o resultado do método pegarSuco();*
- 4) Altere a máquina (fábrica) do quiosque para Ambev.
 - *Imprima novamente o refrigerante e o suco.*

Problema prático

- Responda:

5) Um novo tipo de bebida é distribuído: Água. O que deve ser feito no código?

6) Uma nova empresa de bebidas entrou no mercado e produz apenas sucos. O que deve ser feito no código?

7) A empresa Coca-Cola Company passou a produzir o refrigerante Coca Cola Zero. O que deve ser feito no código?

Referências

- GAMMA, E.; HELM, R.; JOHNSON, R.; VLISSIDES, J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional, 1995. Capítulo 3.