Design Patterns

REVIEW

INTRODUÇÃO

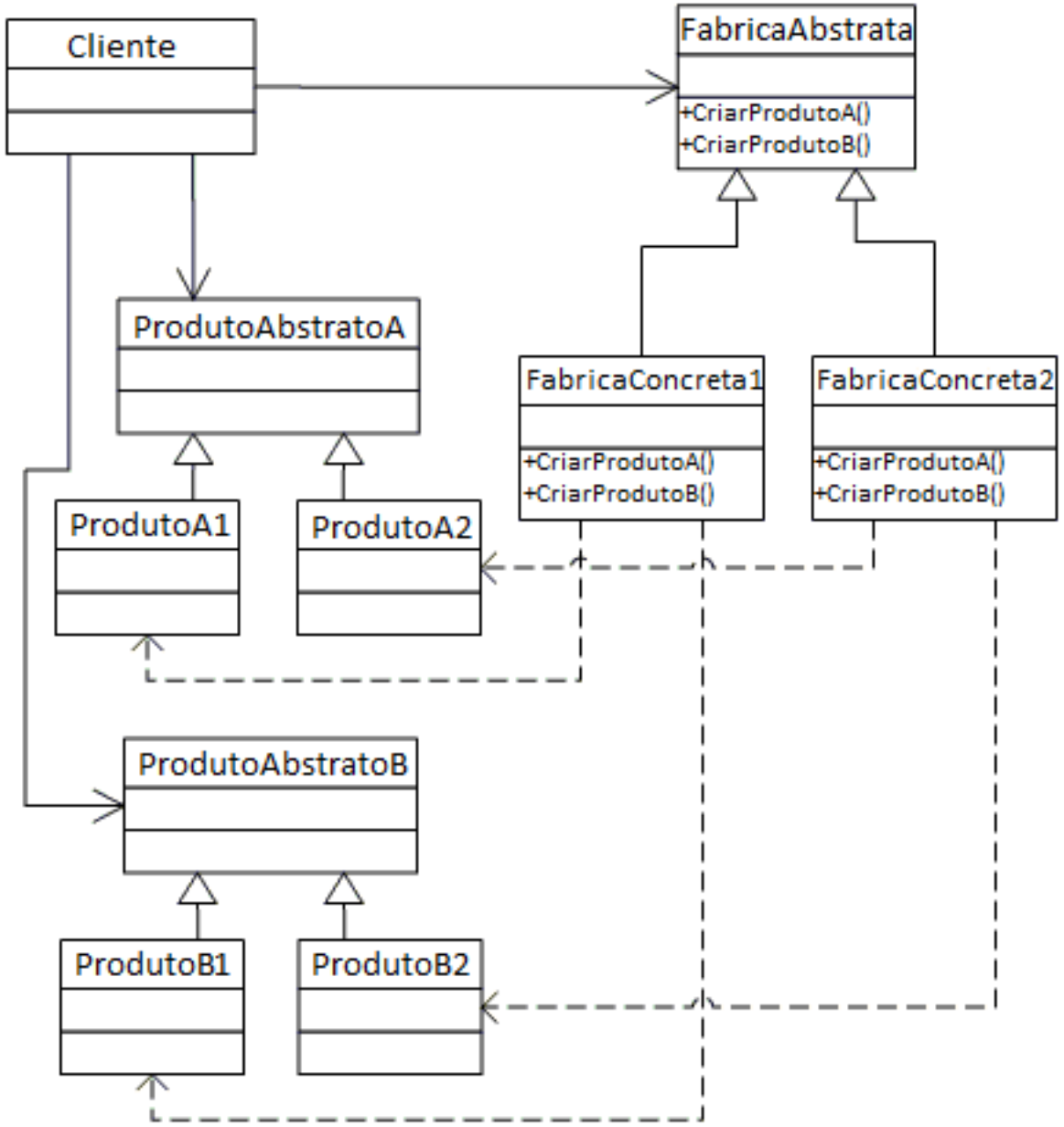
Este padrão permite a criação de famílias de objetos relacionados ou dependentes por meio de uma única interface e sem que a classe concreta seja especificada.

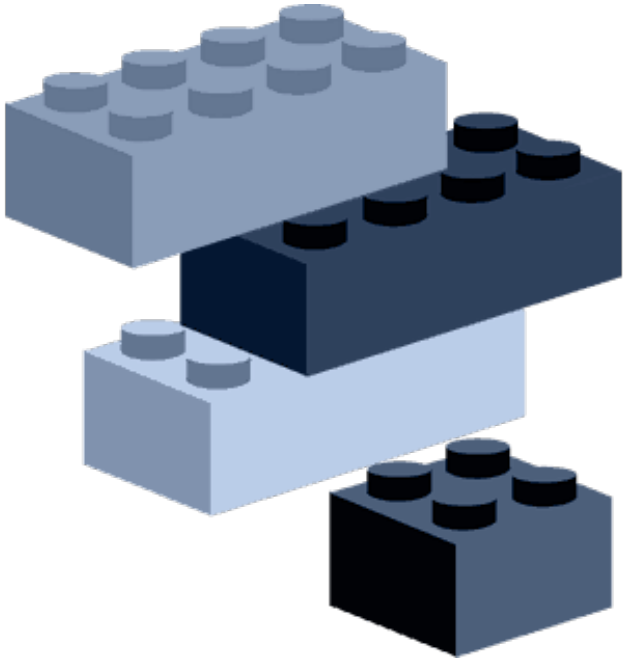
O objetivo é isolar a criação de objetos de seu uso e criar famílias de objetos relacionados sem ter que depender de suas classes concretas.

UNIVAS

CARACTERÍSTICAS

- É um padrão de projeto de criação, ou seja, lida com a criação de objetos;
- É uma fábrica, assim como o Factory Method, e geralmente é composto por múltiplos Factory Metods;
- Visa agrupar famílias de produtos compatíveis, criando uma fábrica concreta por grupo de objetos compatíveis;
- Separa código que cria do que usa;
- Permite a fácil implementação de novas famílias de objetos;
- Toda a programação fica focada nas interfaces, ou classes abstratas.


Cliente
FabricaAbstrata
+CriarProdutoA()
+CriarProdutoB()
ProdutoAbstratoA
FabricaConcreta1
+CriarProdutoA()
+CriarProdutoB()
FabricaConcreta2
+CriarProdutoA()
+CriarProdutoB()
ProdutoA1
ProdutoA2
ProdutoAbstratoB
ProdutoB1
ProdutoB2



ABSTRACT FACTORY

APRESENTAÇÃO E PRÁTICA DO PP

INTRODUÇÃO

O Prototype é um padrão de projeto criacional que permite a criação de novos objetos a partir de um modelo original ou protótipo que é clonado.

Em linhas gerais, ele permite que as subclasses não precisem trabalhar como o Abstract Factory e evite criar objetos utilizando o new, passando a utilizar um método clone().

UNIVAS

CARACTERÍSTICAS

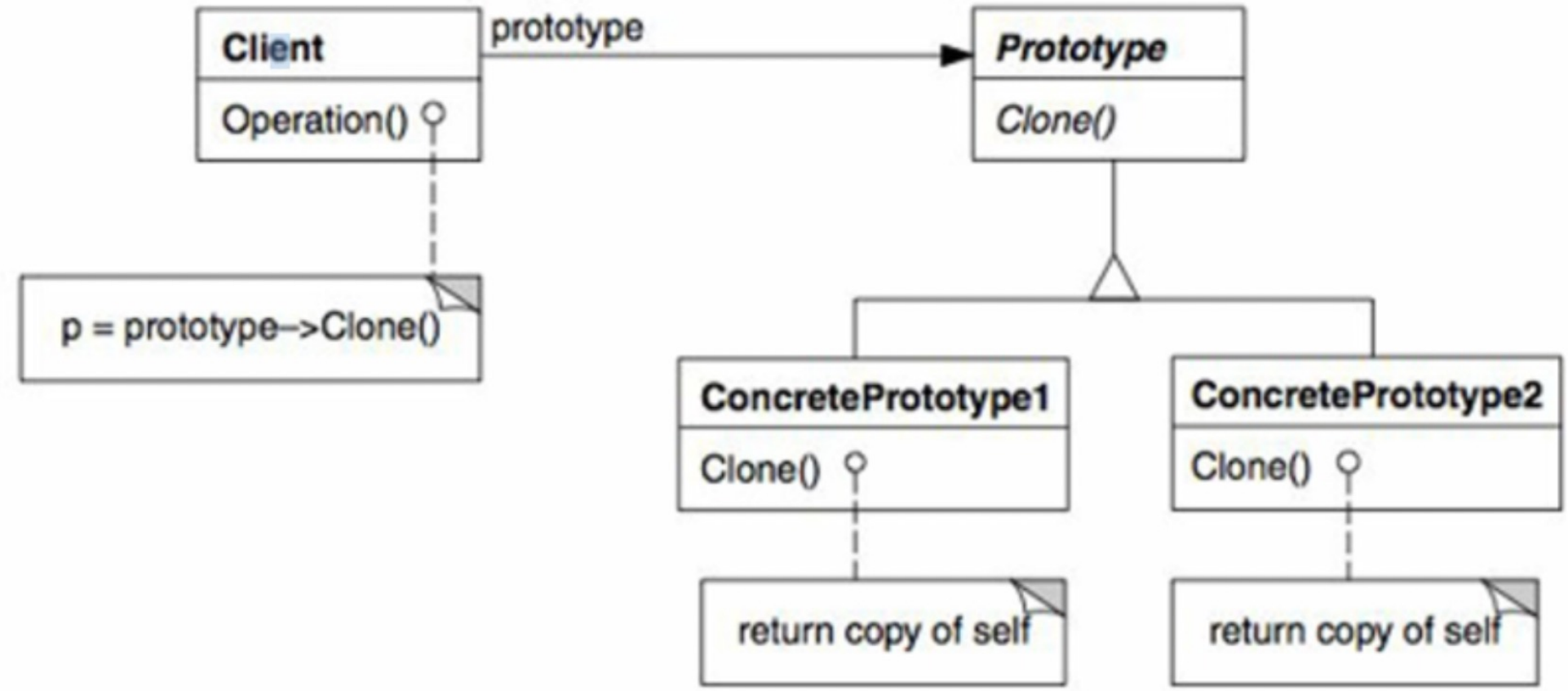
- É um padrão de projeto de criação, ou seja, lida com a criação de objetos;
- O tipo de objeto a ser criado é determinado pelo protótipo;
- É tipicamente usado para evitar a recriação de objetos complexos;
- Ajuda a evitar a explosão de subclasses;
- Utiliza um método clone();
- Evita que o cliente conheça as classe que criam os objetos.

QUANDO USAR?

- Quando a aplicação precisa criar cópias exatas de algum objeto em tempo de execução;
- Quando os sistemas precisam ser independentes da forma como os seus componentes são criados;
- Quando necessário simplificar a hierarquia de classes determinada por sistemas que usam o Abstract Factory,

UNIVAS

ESTRUTURA



UNIVAS

ESTRUTURA

Prototype: declara uma interface para clonar a si próprio.

ConcretePrototype: implementa uma operação para clonar a si próprio.

Client: cria um novo objeto solicitando a um protótipo que clone a si próprio.

UNIVAS

ESTRUTURA

Em Java...

Cloneable: interface que não possui métodos e é utilizada apenas para indicar que o método clone() pode realizar uma cópia, atributo por atributo, das instâncias de uma classe.

CONSEQUÊNCIAS

Positivas:

Oculta classes concretas do código cliente;

Ajuda na criação de objetos complexos;

Evita a explosão de subclasses.

UNIVAS

CONSEQUÊNCIAS

Negativas:

🚫 Difícil clonar clones.

UNIVAS

Obrigado!

Professor Gustavo Dias
luizdias@univas.edu.br