**Aluno: Tomás de Farias Ribeiro Caldas**

**Aula 01 , Módulo 02**

**Questão:**

Pesquise sobre padrões de projeto e escolha um para apresentar e descrever o seu funcionamento. Além disso, explique quais as vantagens e desvantagens comparados a outros e mostre suas referências.

**Resolução:**

**PADRÕES DE PROJETOS:**

Padrões de projetos podem ser informalmente considerados como aplicação de boas práticas e/ou reuso de repertório de soluções.

Os componentes de um padrão de projeto são :

1- Nome - descreve a essência do padrão

2- Objetivo - descreve como o padrão atua.

3- Problema - descreve o problema

4- Solução - descreve a solução

5- Conseqüências - descreve os benefícios da utilização do padrão.

**Padrão escolhido: Prototype** - O padrão Prototype fornece uma outra maneira de se construir objetos de tipos arbitrários:

- **Intenção:** Especificar os tipos de objetos a serem criados usando uma instância-protótipo e criar novos objetos pela cópia desse protótipo

- **Aplicabiidade:** Use o padrão Prototype quando um sistema tiver que ser independente de como os seus produtos são criados, compostos e representados;

**Vantegens :**

**1.** Acrescenta e remove produtos em tempo de execução. Prototype permite incorporar uma nova classe concreta de produto a um sistema, simplesmente registrando uma instância protótipo com o cliente. Isso é um pouco mais flexível do que outros padrões de criação, porque o cliente pode instalar e remover protótipos em tempo de execução.

**2.** Especifica novos objetos pela variação de valores. Sistemas altamente dinâmicos permitem definir novos comportamentos através da composição de objetos

– por exemplo, pela especificação de valores para as variáveis de um objeto

– e não pela definição de novas classes.

Você efetivamente define novos tipos de objetos pela instanciação das classes existentes e registrando as instâncias como protótipos dos objetos-clientes. Um cliente pode exibir um novo comportamento através da delegação de responsabilidades para o protótipo.

Esse tipo de projeto permite aos usuários definir novas “classes” sem ter que programar. De fato, clonar um protótipo é semelhante a instanciar uma classe. O padrão Prototype pode reduzir grandemente o número de classes que um sistema necessita. No nosso editor musical, uma classe GraphicTool pode criar uma variedade ilimitada de objetos musicais.

**3**. Especifica novos objetos pela variação da estrutura. Muitas aplicações constróem objetos com partes e subpartes. Por exemplo, editores para o projeto de circuitos que constroem circuitos a partir de subcircuitos.1 Por questões de conveniência, tais aplicações freqüentemente permitem instanciar estruturas complexas, definidas pelo usuário, para, por exemplo, usar um subcircuito específico repetidas vezes. O padrão Prototype também suporta isso. Simplesmente adicionamos esse subcircuito como um protótipo à paleta dos elementos de circuitos disponíveis. Contanto que o objeto-circuito composto implemente um clone por replicação (deep copy), circuitos com diferentes estruturas podem ser protótipos.

**4**. Reduz o número de subclasses. O Factory Method (112) freqüentemente produz uma hierarquia de classes Creator paralela à hierarquia de classes do produto. O padrão Prototype permite clonar um protótipo em vez de pedir a um método fábrica para construir um novo objeto. Daí não necessitar-se de nenhuma hierarquia de classes Creator. Esse benefício se aplica primariamente a linguagens como C++, que não tratam as classes como objetos de primeira classe. As linguagens que assim o fazem, como Smalltalk e Objective C, obtêm menos benefícios, uma vez que sempre se usa um objeto-classe como um criador. Objetos-classe já funcionam como protótipos nessas linguagens.

**5**. Configura dinamicamente uma aplicação com classes. Alguns ambientes de tempo de execução permitem carregar classes dinamicamente numa aplicação. O padrão Prototype é a chave para a exploração de tais possibilidades numa linguagem como C++. Uma aplicação que quer criar instâncias de uma classe dinamicamente carregada não será capaz de referenciar o seu constructor estaticamente. Em vez disso, o ambiente de tempo de execução cria uma instância de cada classe automaticamente, quando carregada, e registra a instância junto a um gerenciador de protótipo (ver a seção Implementação). Então, a aplicação pode solicitar ao gerenciador de protótipos instâncias de classes recémcarregadas, classes essas que originalmente não estavam “linkadas” ao programa. O framework de aplicações da ET++ [WGM88] tem um sistema de tempo de execução que usa este esquema.

**Desvantagens :** O principal ponto fraco do padrão Prototype é que cada subclasse de Prototype deve implementar a operação Clone, o que pode ser difícil. Por exemplo, acrescentar Clone é difícil quando as classes consideradas já existem. A implementação de Clone pode ser complicada quando uma estrutura interna dessas classes inclui objetos que não suportam operação de cópia ou têm referências circulares

Referencias:

- <https://www.macoratti.net/vb_pd1.htm>

-<https://web.inf.ufpr.br/didonet/wp-content/uploads/sites/12/2021/03/Padroes-de-projeto.pdf>