**UNIVALE – UNIVERSIDADE DO VALE DO RIO DOCE**

**FATEC – FACULDADE DE CIENCIAS TECNOLOGICAS**

**ENGENHARIA DE SOFTWARE II**

**PADRÕES DE PROJETO**

**Grupo:**

**Bryan França, Charles Andrade, Farley Souza, Israel Zequinelli, Ulisses Silveira**

**Governador Valadares, MG**

**Outubro 2018**

**História dos padrões de projeto**

O padrão de projeto surgiu na década de 80 com o arquiteto e professor Christopher Alexandre, no seu livro “A Times Way of Building” (Oxford University Press, 1979). No livro ele estabelece que um padrão deve ter as seguintes características:

*Encapsulamento*: um padrão encapsula um problema ou solução bem definida. Ele deve ser independente, específico e formulado de maneira a ficar claro onde ele se aplica.

*Generalidade*: todo padrão deve permitir a construção de outras realizações a partir deste padrão.

*Equilíbrio*: quando um padrão é utilizado em uma aplicação, o equilíbrio dá a razão, relacionada com cada uma das restrições envolvidas, para cada passo do projeto. Uma análise racional que envolva uma abstração de dados empíricos, uma observação da aplicação de padrões em artefatos tradicionais, uma série convincente de exemplos e uma análise de soluções ruins ou fracassadas pode ser a forma de encontrar este equilíbrio.

*Abstração*: os padrões representam abstrações da experiência empírica ou do conhecimento cotidiano.

*Abertura*: um padrão deve permitir a sua extensão para níveis mais baixos de detalhe.

*Combinatoriedade*: os padrões são relacionados hierarquicamente. Padrões de alto nível podem ser compostos ou relacionados com padrões que endereçam problemas de nível mais baixo.

Além da definição das características de um padrão, Alexander definiu o formato que a descrição de um padrão deve ter. Ele estabeleceu que um padrão deve ser descrito em cinco partes:

*Nome*: uma descrição da solução, mais do que do problema ou do contexto;

*Exemplo*: uma ou mais figuras, diagramas ou descrições que ilustrem um protótipo de aplicação;

*Contexto*: a descrição das situações sob as quais o padrão se aplica;

*Problema*: uma descrição das forças e restrições envolvidos e como elas interagem;

*Solução*: relacionamentos estáticos e regras dinâmicas descrevendo como construir artefatos de acordo com o padrão, frequentemente citando variações e formas de ajustar a solução segundo as circunstâncias. Inclui referências a outras soluções e o relacionamento com outros padrões de nível mais baixo ou mais alto.

Em 1987, a partir dos conceitos criados por Alexander, os programadores Kent Beck e Ward Cunningham propuseram os primeiros padrões de projeto para a área da ciência da computação. Porém, o movimento ao redor de padrões de projeto só ganhou popularidade em 1995 quando foi publicado o livro Design Patterns: Elements of Reusable Object-Oriented Software. Os autores desse livro, Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides, são conhecidos como a "Gangue dos Quatro" (Gang of Four) ou simplesmente "GoF". Segundo o livro, os padrões são divididos em 23 tipos e devido a quantidade de padrões houve a necessidade de dividi-los em três categorias: de Criação, Estrutural e Comportamental. Todas essas categorias trabalham com os escopos de classe e objetos, o escopo de classe tem como característica a utilização de herança para garantir a flexibilidade na manipulação das subclasses. Já os padrões com escopo de Objeto, delegam para um objeto (no caso o protótipo) a responsabilidade de instanciar novos objetos.

**Categorias de padrões de projeto**

Os padrões de criação têm como intenção principal abstrair o processo de criação de objetos, ou seja, a sua instanciação. Desta maneira o sistema não precisa se preocupar com questões sobre, como o objeto é criado, como é composto, qual a sua representação real. Quando se diz que o sistema não precisa se preocupar com a instanciação do objeto quer dizer que, se ocorrer alguma mudança neste ponto, o sistema em geral não será afetado. São os padrões de criação: Factory Method, Abstract Factory, Builder, Prototype, sigleton.

Os padrões estruturais vão se preocupar em como as classes e objetos são compostos, ou seja, como é a sua estrutura. O objetivo destes padrões e facilitar o design do sistema identificando maneiras de realizar o relacionamento entre as entidades, deixando o desenvolvedor livre desta preocupação. São estes os padrões estruturais: Composite, Adapter, Brigde, Decorator, Facade, Proxy, Flyweight.

Os padrões comportamentais atuam sobre como responsabilidades são atribuídas as entidades, ou seja, qual o comportamento das entidades. Estes padrões facilitam a comunicação entre os objetos, distribuindo as responsabilidades e definindo a comunicação interna. Sendo estes os padrões comportamentais: Strategy, Interator, Template Method, Observer, Mediator, Command, Memento, Chain of Responsibility, Interpreter, State, Visitor.