PUC Minas
Virtual

Técnicas de Desenvolvimento

Samuel Almeida Cardoso

PUC Minas
Virtual

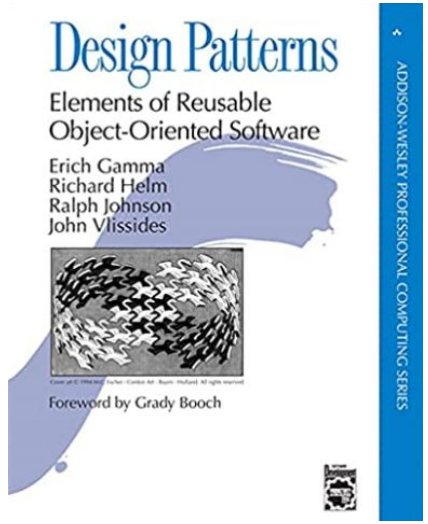
Arquiteturas de Aplicações e Soluções

Padrões de Design

- São soluções repetíveis para um problema comum no projeto de software e de projeto. Um padrão de design não é uma regra aplicável em qualquer lugar, ele deve ser transformado via código e adaptado para a realidade do projeto. É uma descrição ou modelo de como resolver um problema que pode ser usado em muitas situações diferentes.

GoF

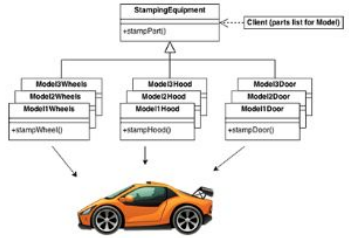
- Padrões de Criação
- Padrões Estruturais
- Padrões Comportamentais



Padrões de Design - Padrões de Criação

Creational design patterns

These design patterns are all about class instantiation. This pattern can be further divided into class-creation patterns and object-creational patterns. While class-creation patterns use inheritance effectively in the instantiation process, object-creation patterns use delegation effectively to get the job done.



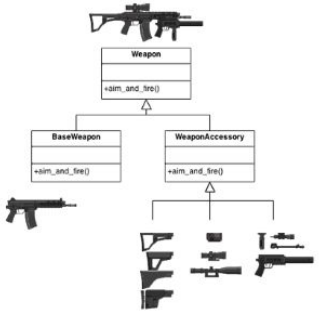
- **Abstract Factory**
  Creates an instance of several families of classes
- **Builder**
  Separates object construction from its representation
- **Factory Method**
  Creates an instance of several derived classes
- **Object Pool**
  Avoid expensive acquisition and release of resources by recycling objects that are no longer in use
- **Prototype**
  A fully initialized instance to be copied or cloned
- **Singleton**
  A class of which only a single instance can exist

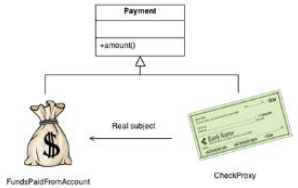
Padrões de Design - Padrões Estruturais

Structural design patterns

These design patterns are all about Class and Object composition. Structural class-creation patterns use inheritance to compose interfaces. Structural object-patterns define ways to compose objects to obtain new functionality.

- **Adapter**
  Match interfaces of different classes
- **Bridge**
  Separates an object's interface from its implementation
- **Composite**
  A tree structure of simple and composite objects
- **Decorator**
  Add responsibilities to objects dynamically
- **Facade**
  A single class that represents an entire subsystem
- **Flyweight**
  A fine-grained instance used for efficient sharing
- **Private Class Data**
  Restricts accessor/mutator access
- **Proxy**
  An object representing another object



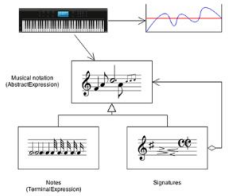


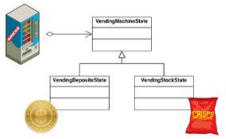
Padrões de Design - Padrões Comportamentais

Behavioral design patterns

These design patterns are all about Class's objects communication. Behavioral patterns are those patterns that are most specifically concerned with communication between objects.

- **Chain of responsibility**
  A way of passing a request between a chain of objects
- **Command**
  Encapsulate a command request as an object
- **Interpreter**
  A way to include language elements in a program
- **Iterator**
  Sequentially access the elements of a collection
- **Mediator**
  Defines simplified communication between classes
- **Memento**
  Capture and restore an object's internal state
- **Null Object**
  Designed to act as a default value of an object
- **Observer**
  A way of notifying change to a number of classes
- **State**
  Alter an object's behavior when its state changes
- **Strategy**
  Encapsulates an algorithm inside a class
- **Template method**
  Defer the exact steps of an algorithm to a subclass
- **Visitor**
  Defines a new operation to a class without change





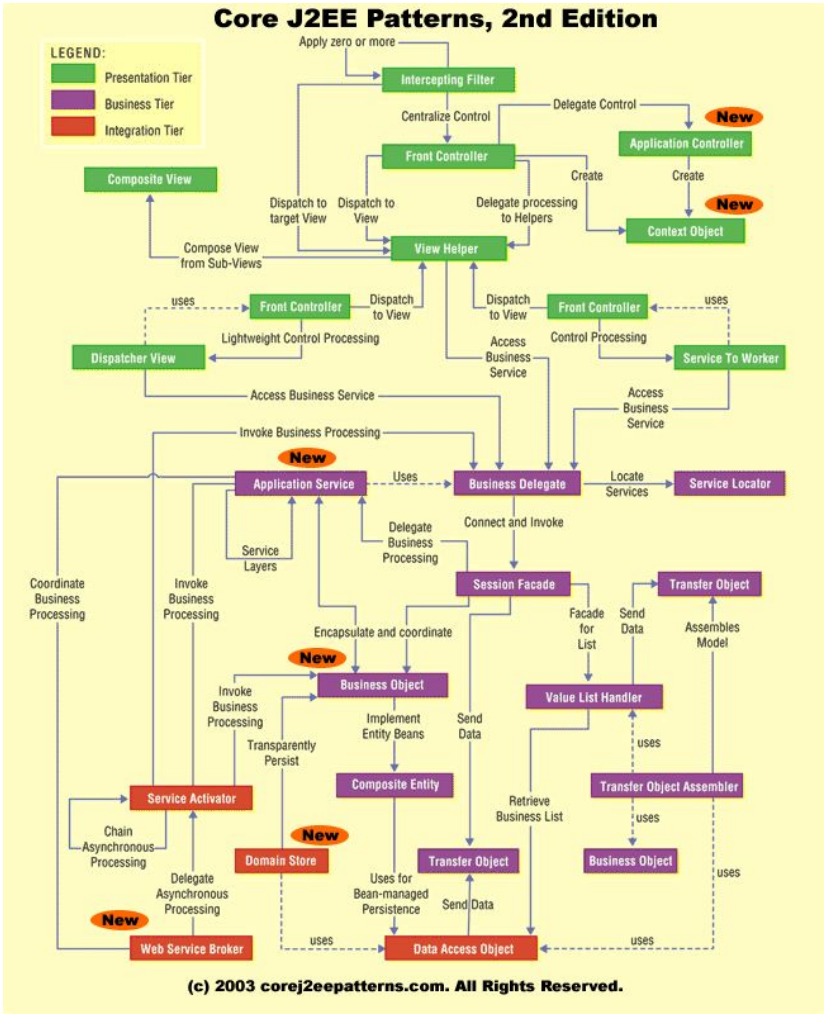
Padrões de Design

https://refactoring.guru/pt-br/design-patterns

Padrões de Design para a Plataforma JEE

- Padrões de Apresentação
- Padrões de Negócio
- Padrões para Camadas de Integração

http://www.corej2eepatterns.com

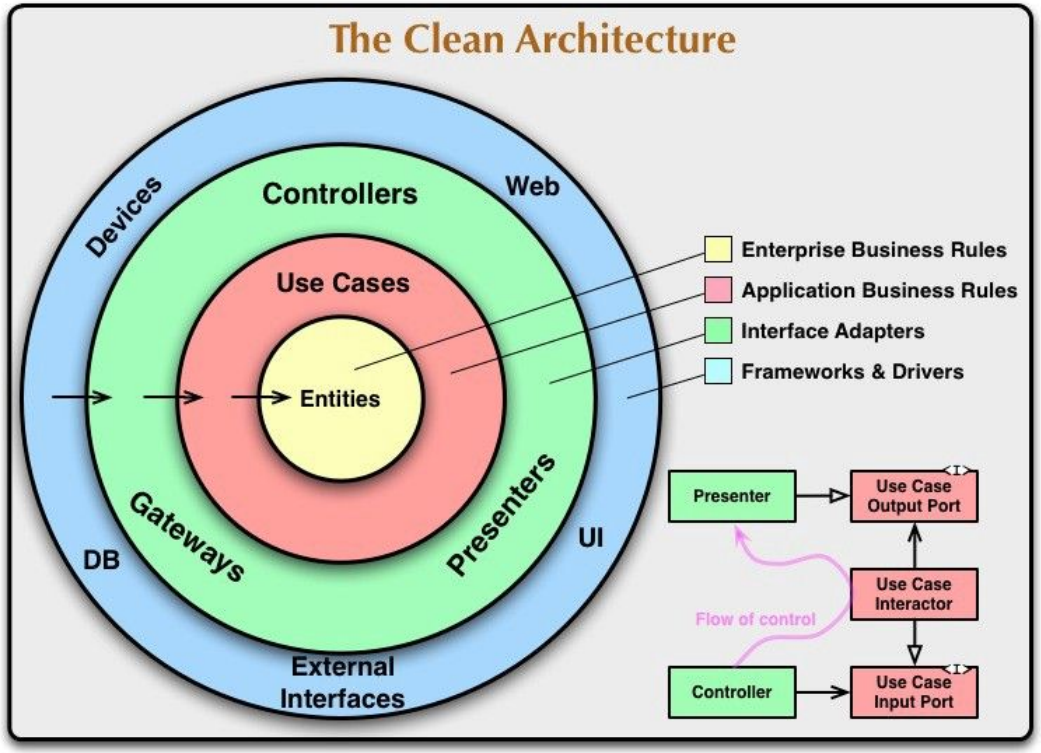


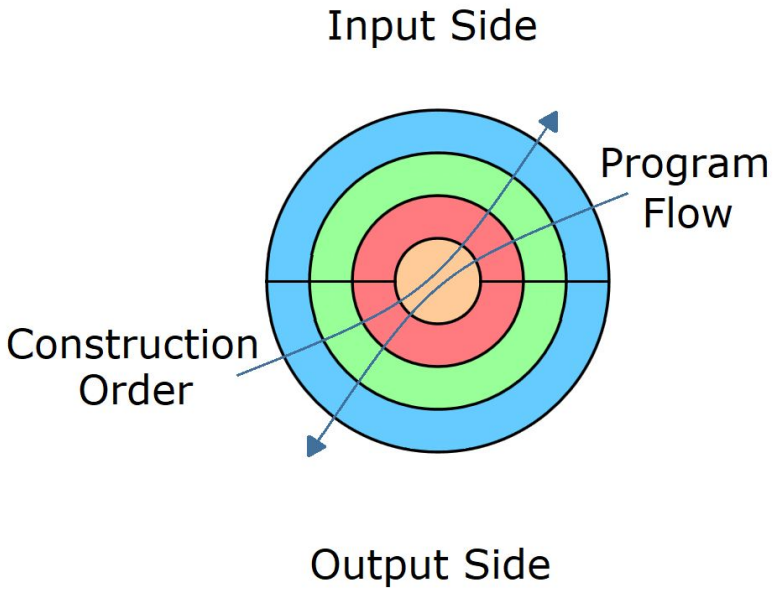
Outros padrões e referências conceituadas

- Arquitetura Limpa
- Arquitetura Hexagonal
- Arquitetura Onion
- Arquitetura em Camadas

http://cleancoder.com

Arquitetura Limpa





PUC Minas
Virtual