

UML e Desenvolvimento Java

Diagrama de Pacotes & Padrão de Projeto Builder


Gustavo de Miranda Gonçalves

gustavo.miranda@prof.infnet.edu.br

UML
Diagrama de Pacotes

Tipos de Diagramas do UML

UML 2 possui 14 tipos de diagramas divididos em duas categorias:


- **Diagramas Estruturais**  Enfatizam os elementos que precisam estar presentes no sistema modelado.
- **Diagramas Comportamentais**
Enfatizam o que precisa acontecer no sistema modelado.

Tipos de Diagramas do UML

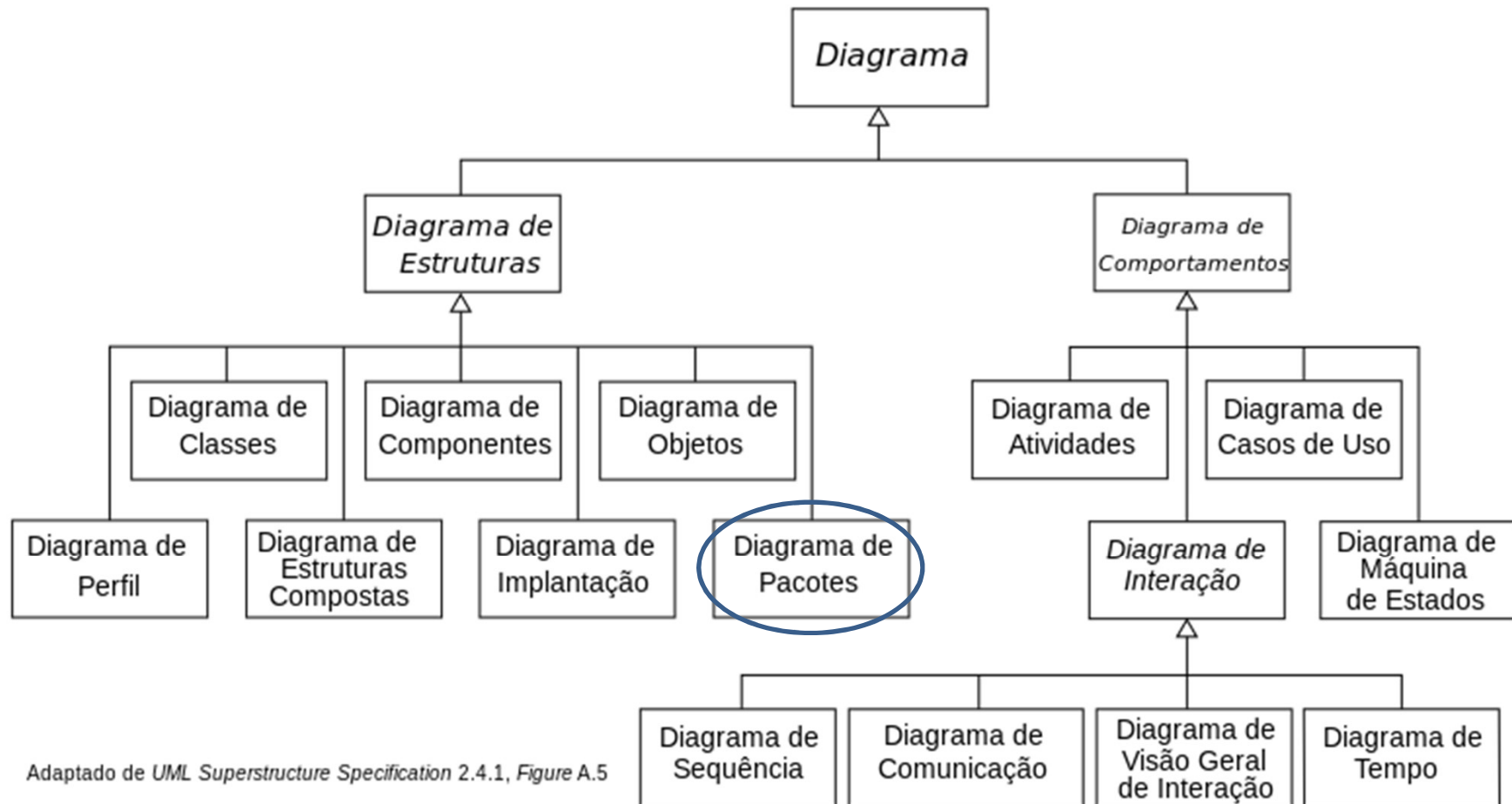
UML 2 possui 14 tipos de diagramas divididos em duas categorias:

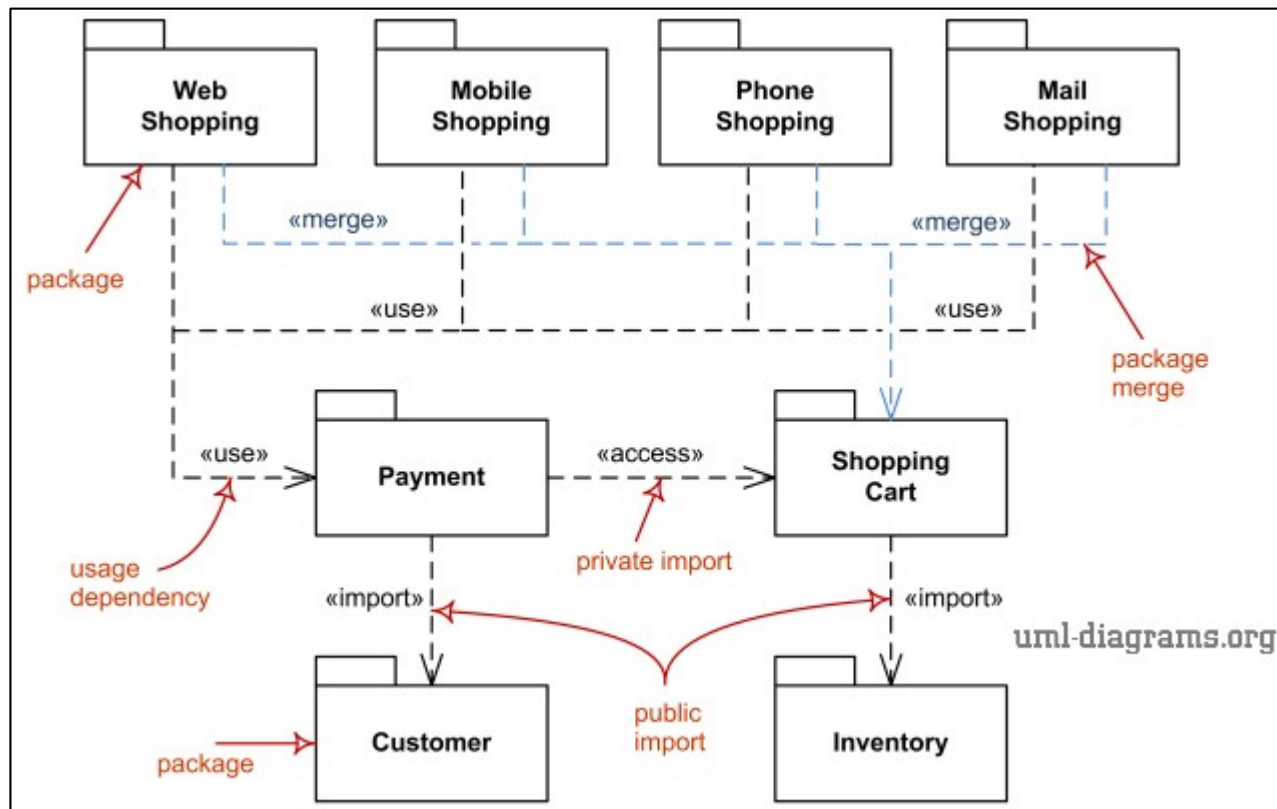
- **Diagramas Estruturais**

Enfatizam os elementos que precisam estar presentes no sistema modelado.

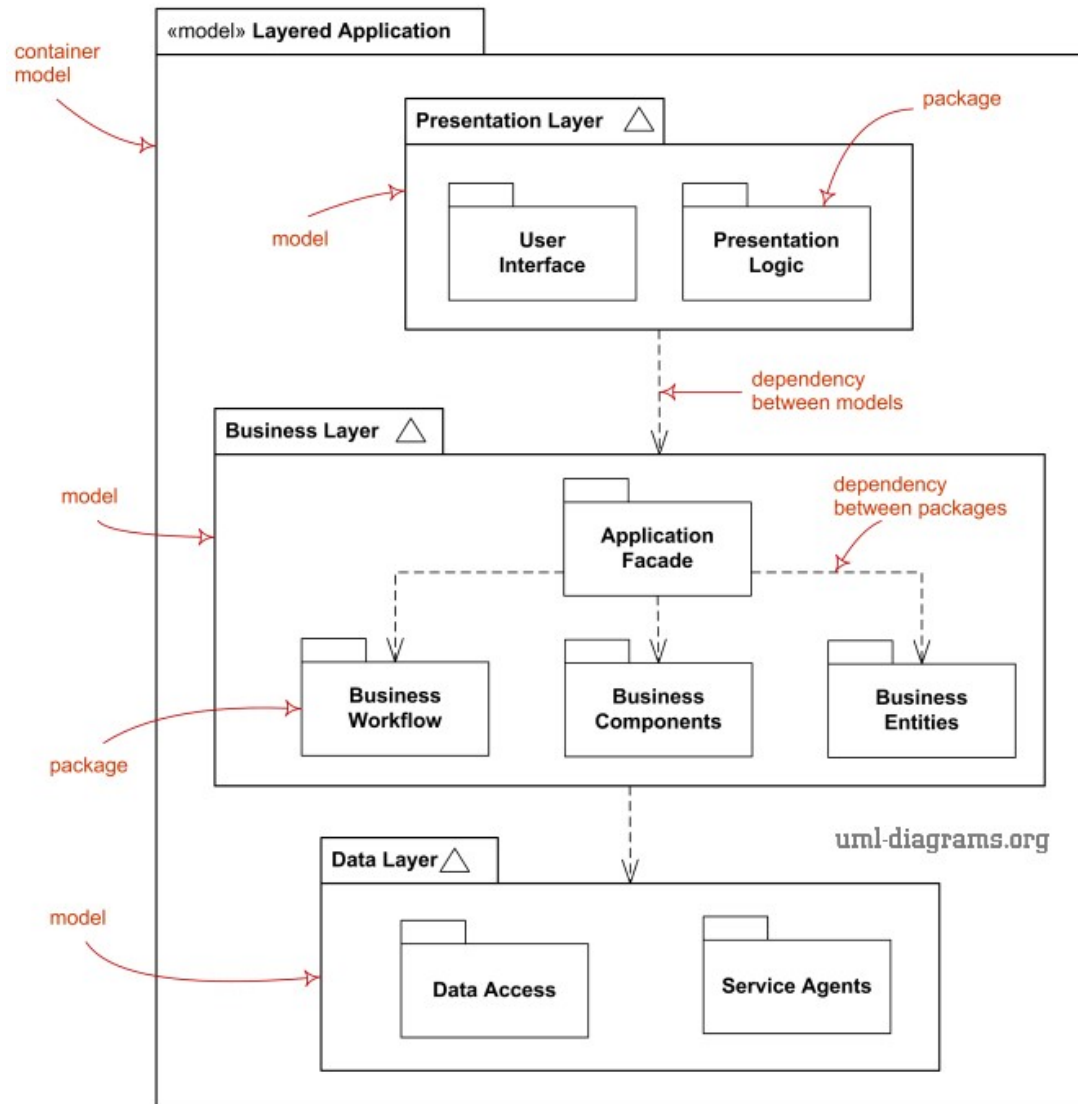
- Diagrama de Classes
- Diagrama de Objetos
- Diagrama de Componentes
- Diagrama de Instalação ou de Implantação
- Diagrama de Pacotes 
- Diagrama de Estrutura Composta
- Diagrama de Perfil

Tipos de Diagramas do UML





<http://www.uml-diagrams.org/package-diagrams-overview.html>



<http://www.uml-diagrams.org/package-diagrams-overview.html>



UML: Diagrama de Pacotes

O que é um Diagrama de Pacotes?

Descreve uma funcionalidade proposta para um novo sistema que será projetado.
É uma ferramenta útil para o levantamento de requisitos funcionais.

Um tipo de diagrama de contexto que apresenta os elementos externos de um sistema e as maneiras segundo as quais eles as utilizam.

Design Pattern *Builder*

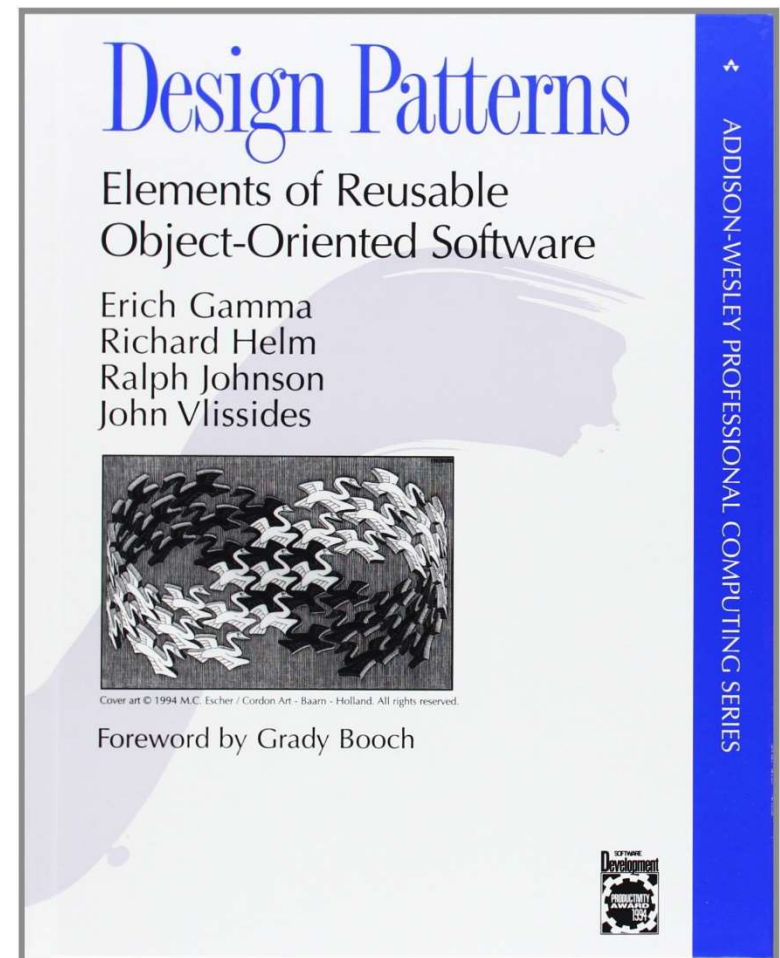


Padrões de Projeto (*Design Patterns*)

- **Padrões GOF**
Gang of Four

Design Patterns: Tipos de Padrões GOF (Gang of Four)

- Padrões de Criação
- Padrões Estruturais
- Padrões Comportamentais



Design Patterns: Tipos de Padrões GOF (Gang of Four)

Criação	Estrutural	Comportamental	
Abstract Factory	Adapter	Chain of Responsibility	State
➔ Builder	Bridge	Command	Strategy
Factory Method	Composite	Interpreter	Template Method
Prototype	Decorator	Iterator	Visitor
Singleton	Facade	Mediator	
	Flyweight	Memento	
	Proxy	Observer	

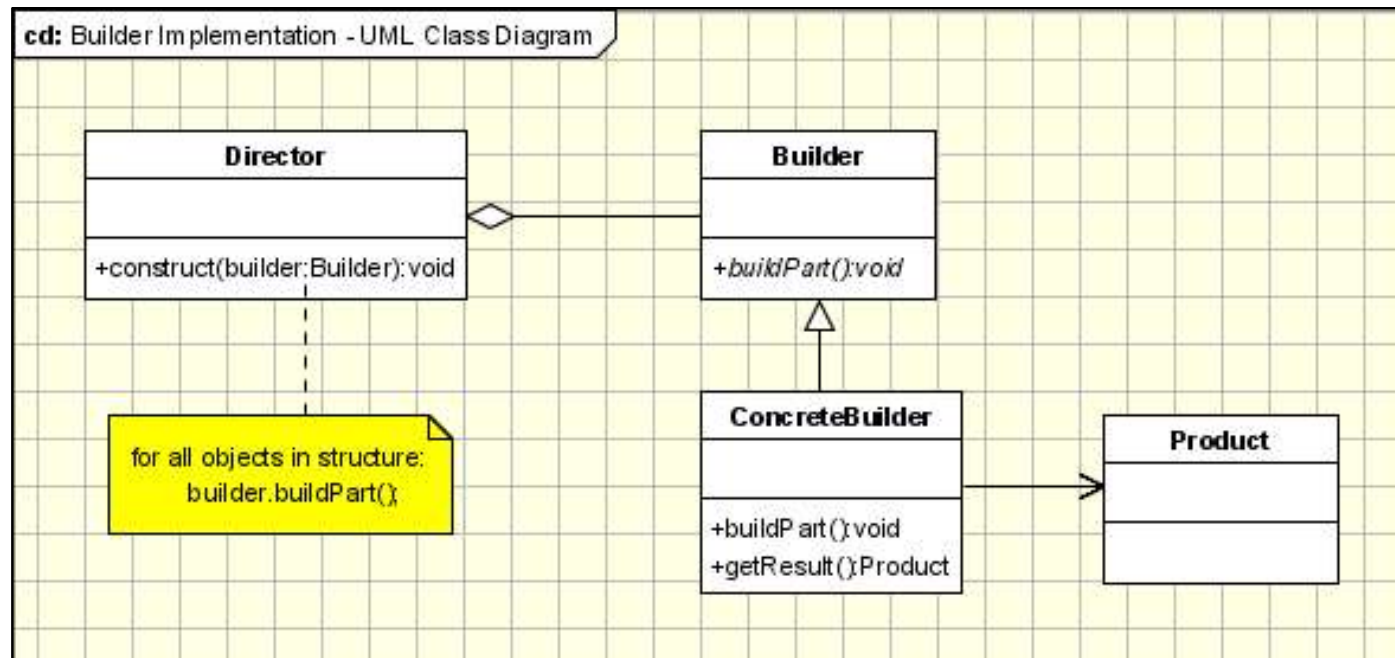


*Design Patterns: **Builder***

Quando deve ser usado?

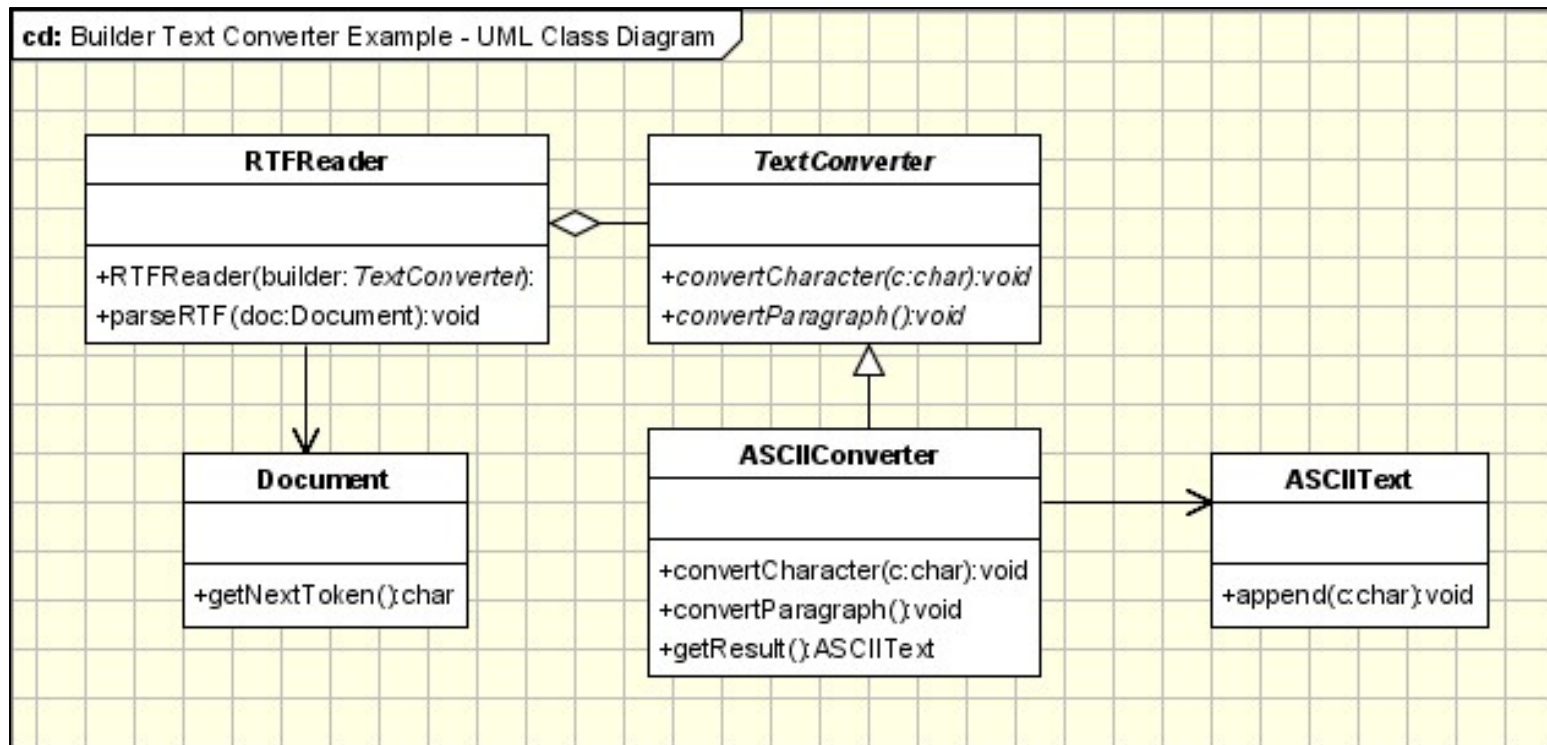
- Quando o algoritmo de criação de um objeto complexo é independente das que compõem o objeto.
- Quando o sistema precisa possibilitar diferentes representações para os objetos que estão sendo construídos.

Design Patterns: **Builder**



<http://www.oodeesign.com/builder-pattern.html>

Design Patterns: **Builder**



<http://www.oodeesign.com/builder-pattern.html>