

---


*UML*

*Diagrama de Casos de Uso*

---

## 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

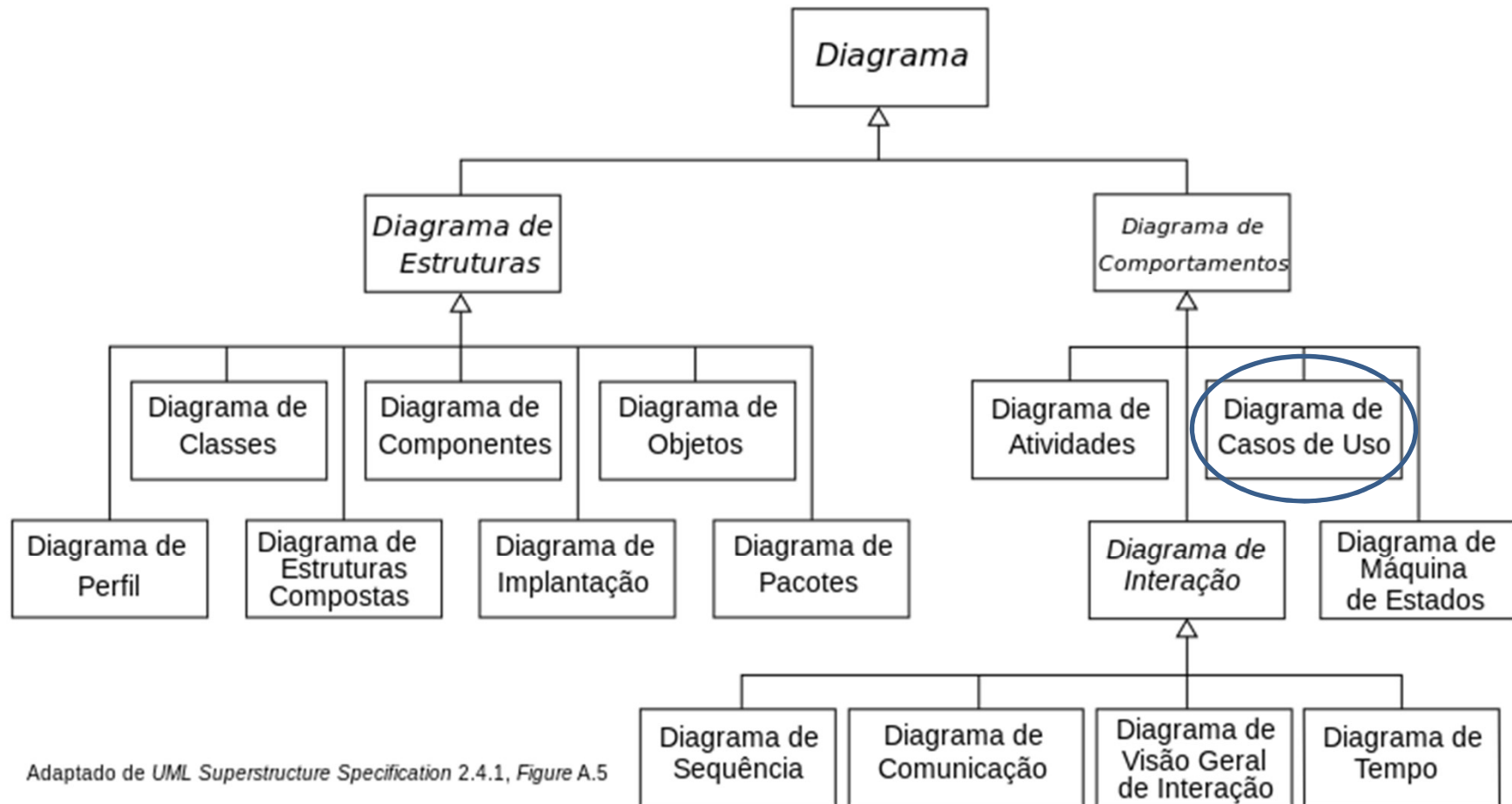
UML 2 possui 14 tipos de diagramas divididos em duas categorias:

- **Diagramas Comportamentais**

Enfatizam o que precisa acontecer no sistema modelado.

- Diagrama de Caso de Uso 
- Diagrama de Transição de Estados (ou de Estados)
- Diagrama de Atividade
- Diagrama de Sequência
- Diagrama Visão Geral de Interação (ou de Interação)
- Diagrama de Colaboração (ou Comunicação)
- Diagrama de Tempo (ou Temporal)

## Tipos de Diagramas do UML





## ***UML: Diagrama de Sequência***

O que é um Diagrama de Caso de Uso?

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.



- Representa todos os casos de uso de um sistema utilizando a linguagem UML.
- Por meio dele é possível visualizar, em um alto nível de abstração, quais os elementos (atores) interagem com o sistema em cada funcionalidade.

- **Ator**

- Humano, dispositivo ou outro software.

- **Generalização**

Um ator pode herdar o papel de outro.

Representação: Linha sólida com um triângulo em direção ao ator mais geral.

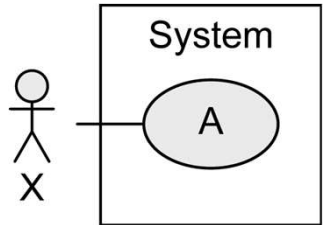


- **Relações entre casos de uso**

- **Incluir** <<include>>

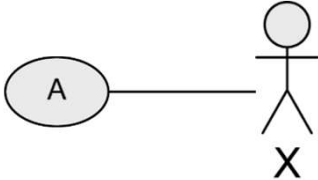

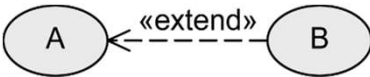
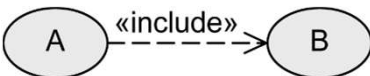
O comportamento do caso de uso incluído é inserido obrigatoriamente no comportamento do caso de uso inclusor.

- **Estender** <<extends>>

O comportamento do caso de uso extensor pode ser ou não inserido no caso de uso estendido.

Nome	Notação	Descrição
Sistema		Limites entre o sistema e os usuários do Sistema.
Use case		Unidade de funcionalidade do Sistema.
Actor		X: Papel do ator

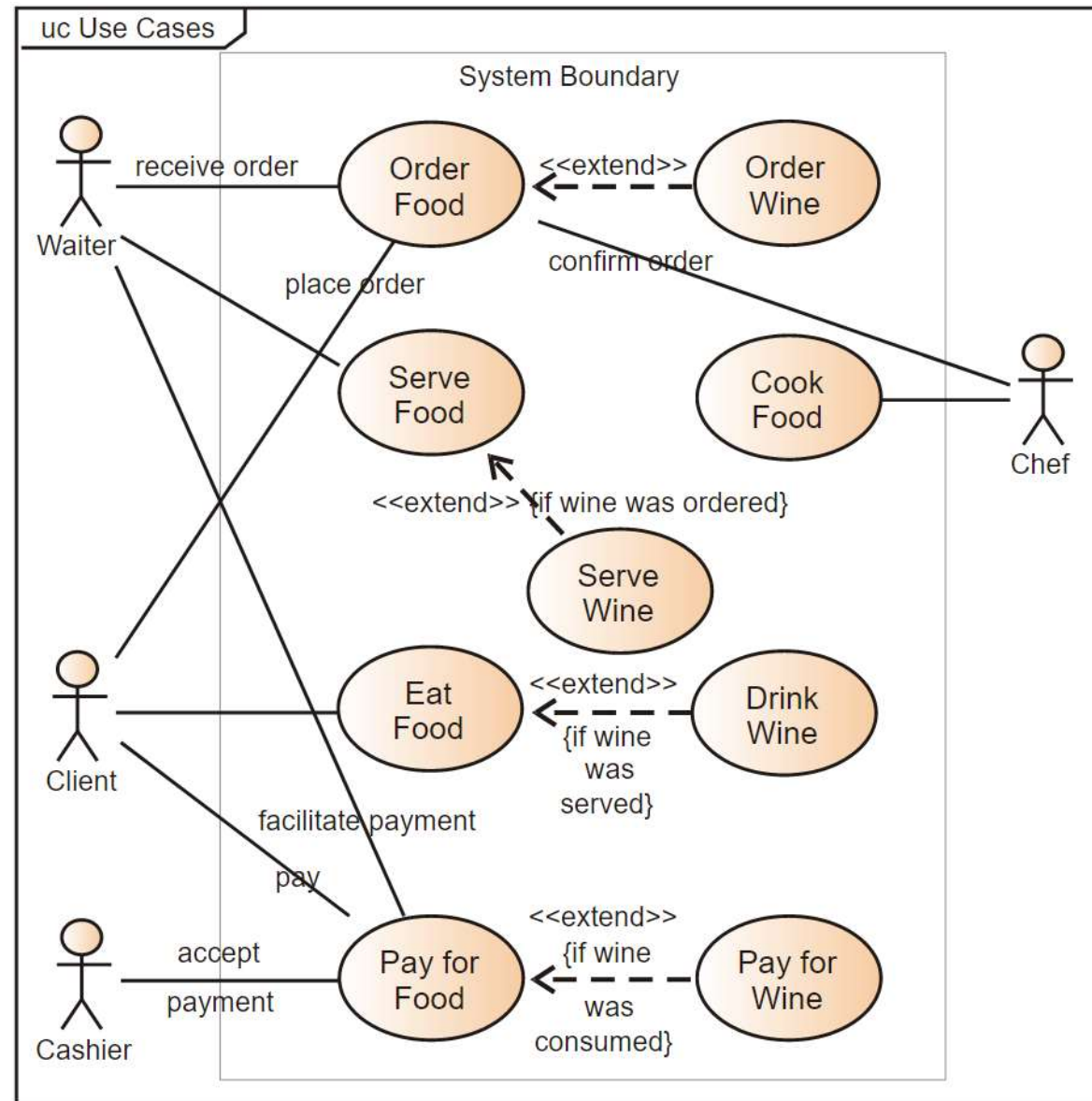


Name	Notation	Description
Associação		Relacionamento entre casos de uso e atores
Generalização		Relacionamento de herança entre casos de uso e atores
Extensão		B estende A: uso opcional do caso de uso B pelo caso de uso A
Inclusão		A inclui B: uso necessário do caso de uso B pelo caso de uso A



Cliente	Sistema
Insere o cartão no caixa eletrônico	
	Apresenta solicitação de senha
Digita a senha	
	Exibe menu de operações disponíveis
Solicita realização de saque	
	Requisita a quantia que será sacada
Retira a quantia e o recibo da transação	

<https://lms.infnet.edu.br/moodle/mod/page/view.php?id=181962>

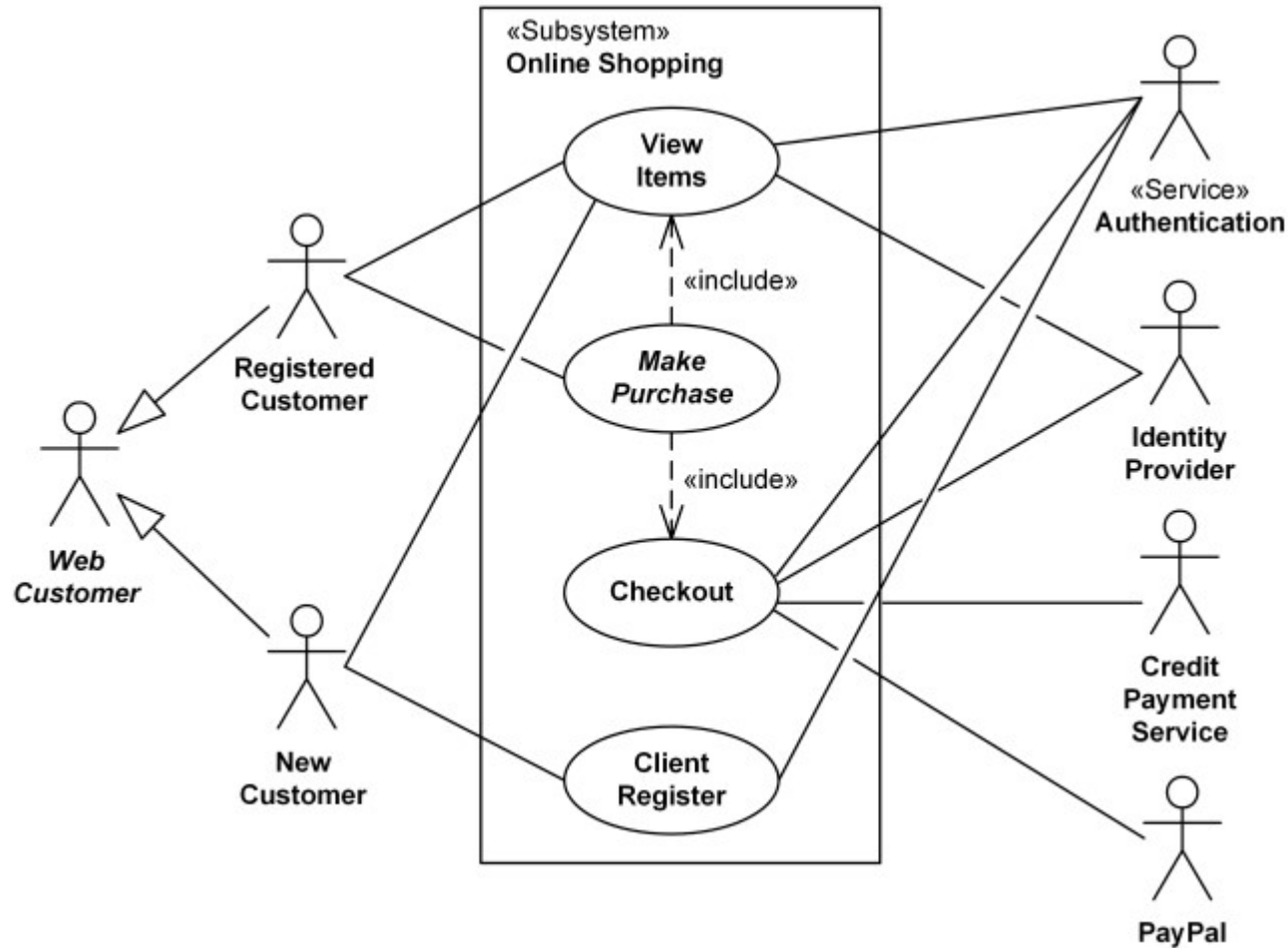


Kishorekumar  
62 (redrawn by Marcel  
Douwe Dekker)

Creative  
Commons Attribution-Share  
Alike 3.0 Unported

## UML: Diagrama de Caso de Uso

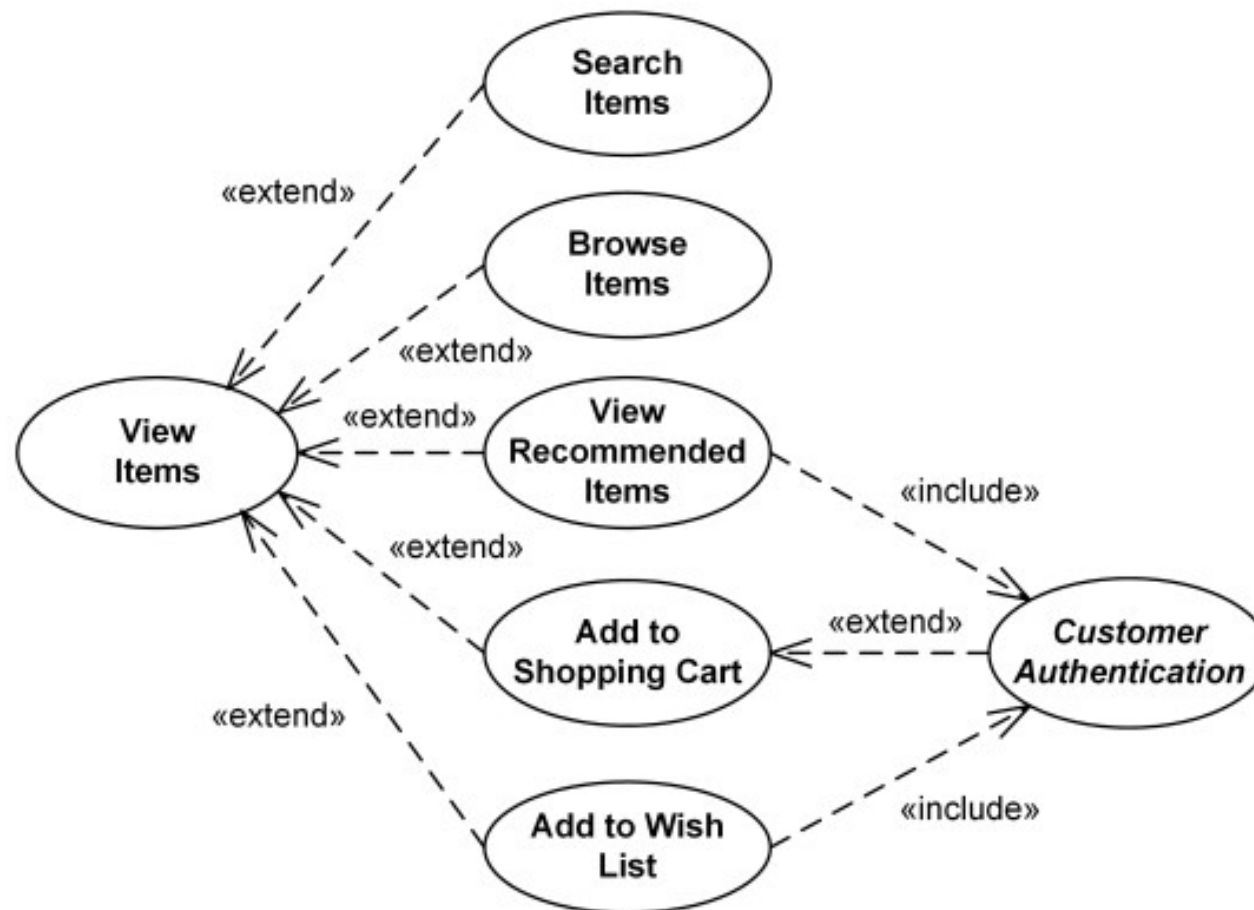
Exemplo Online Store - Top Level



<http://www.uml-diagrams.org/examples/online-shopping-use-case-diagram-example.html?context=uc-examples>

## UML: Diagrama de Caso de Uso

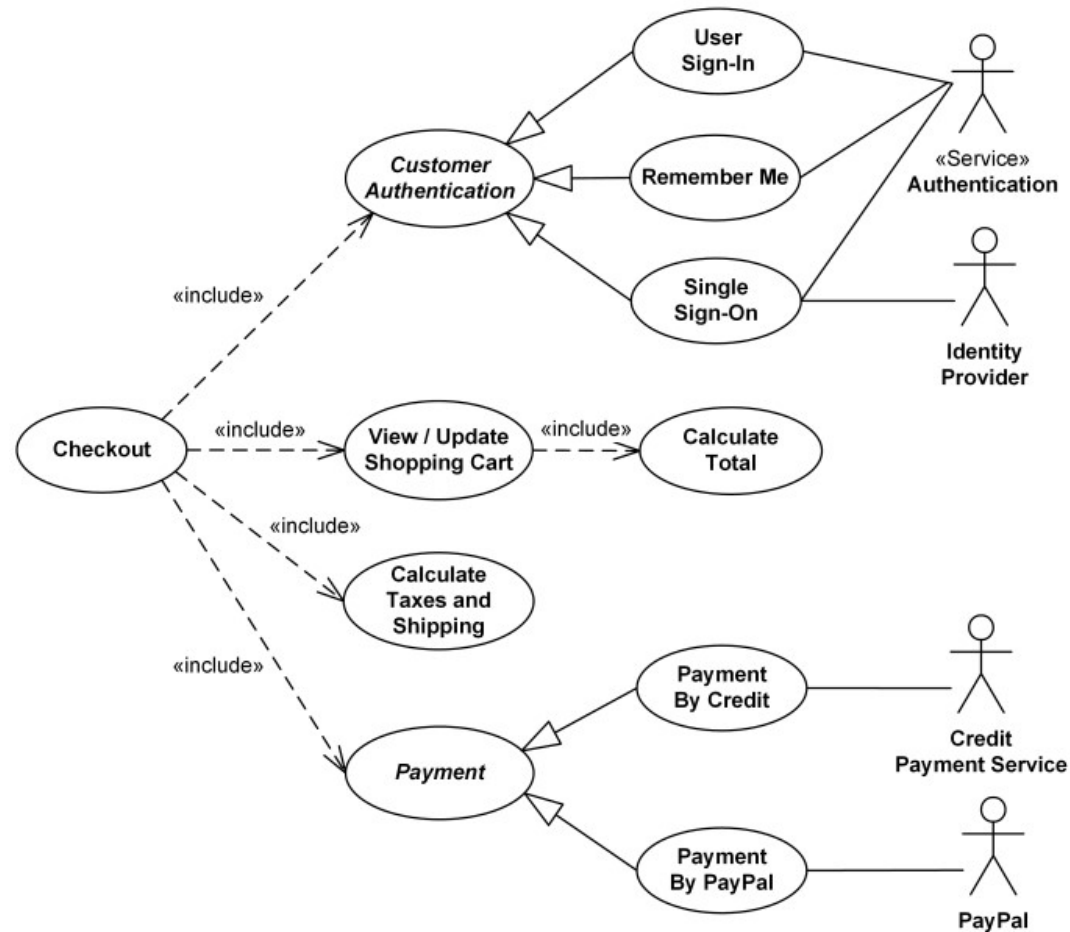
Exemplo Online Store - View Items



<http://www.uml-diagrams.org/examples/online-shopping-use-case-diagram-example.html?context=uc-examples>

## UML: Diagrama de Caso de Uso

Exemplo Online Store – Checkout, authentication and payment use cases



<http://www.uml-diagrams.org/examples/online-shopping-use-case-diagram-example.html?context=uc-examples>

---

*Design Pattern*  
*Abstract Factory*

---



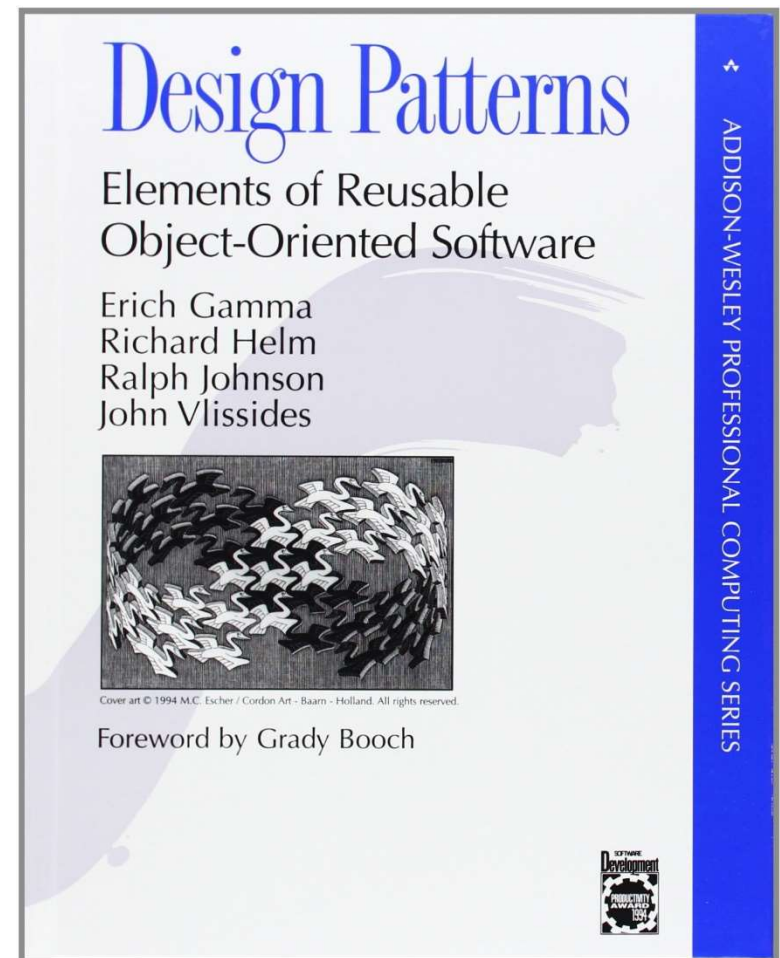
## Padrões de Projeto (*Design Patterns*)

- **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	

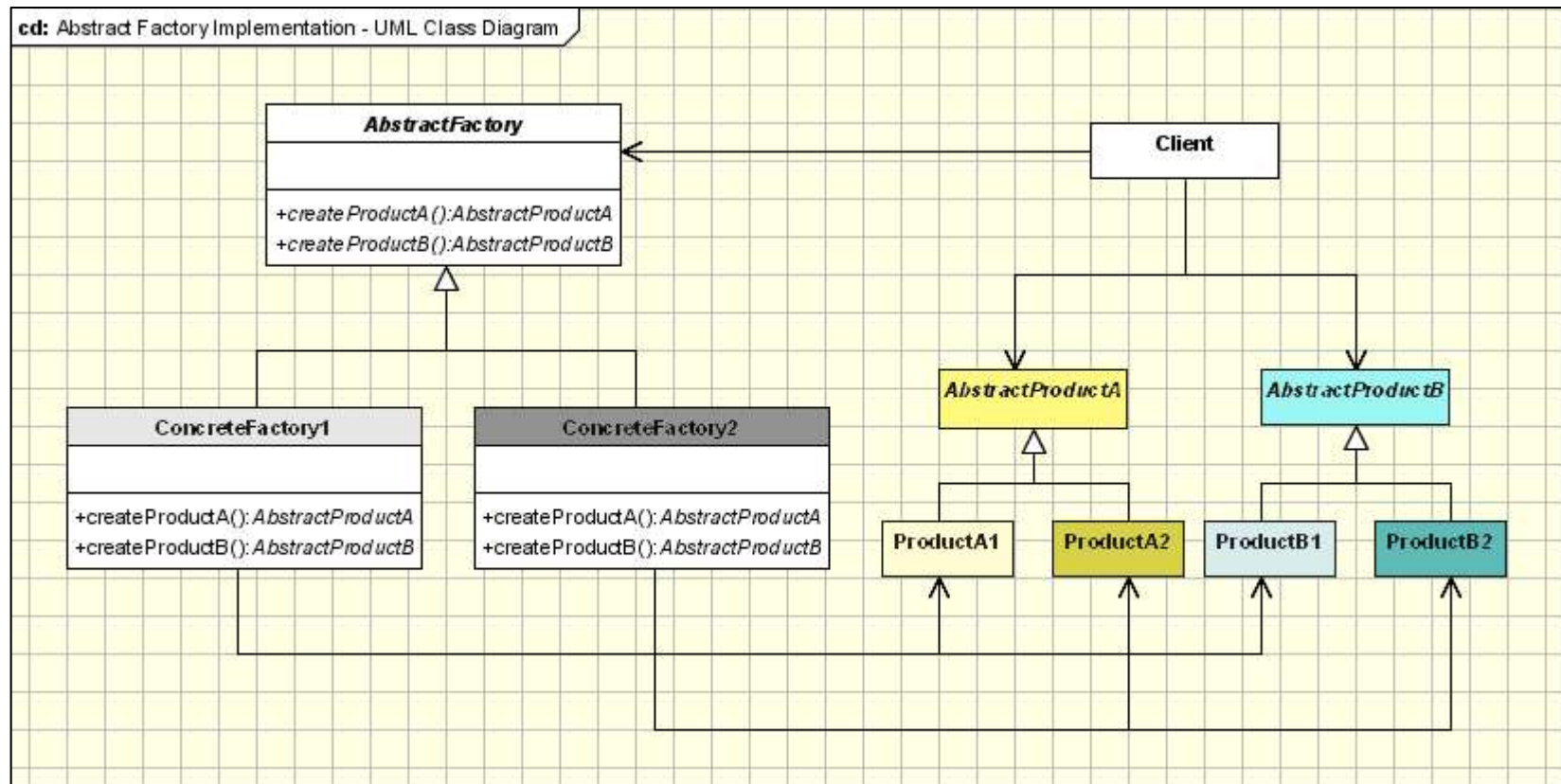


### *Design Patterns: **Abstract Factory***

#### **Quando deve ser usado?**

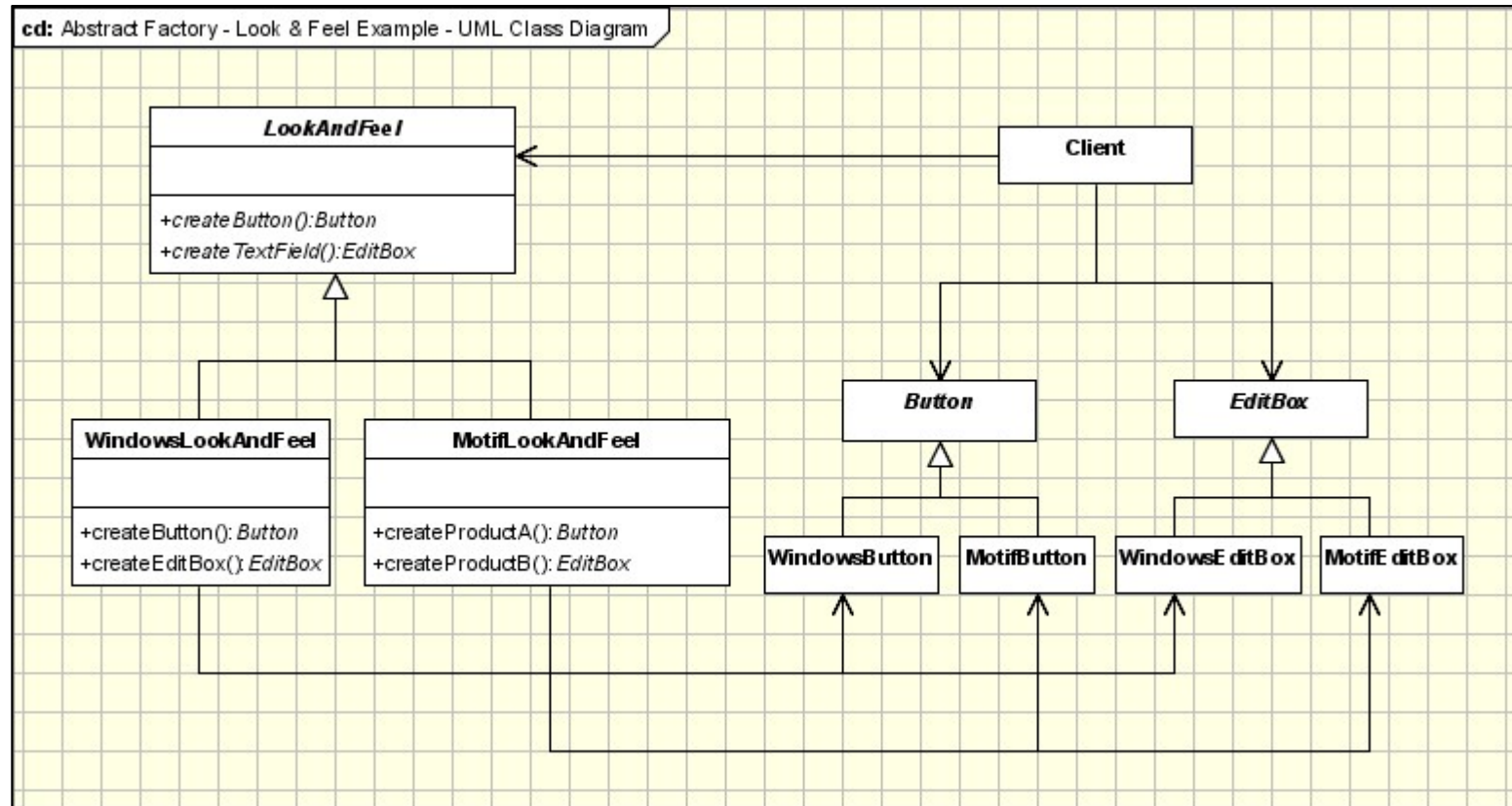
- O sistema precisa ser independente da forma como os produtos com os quais trabalha são criados.
- O sistema é ou deveria ser configurado para trabalhar com múltiplas famílias de produtos.
- A família de produtos é projetada para trabalhar apenas em conjunto.
- A criação de uma biblioteca de produtos é necessária, para cada produto, apenas a interface é relevante, não a implementação.

## Design Patterns: **Abstract Factory**



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

## Design Patterns: **Abstract Factory**



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