ATIVIDADES: CONCEITOS DE MODELOS

A modelagem é o processo de criar representações simplificadas e estruturadas de sistemas complexos, com o objetivo de compreender, analisar, projetar, documentar e comunicar seus componentes e comportamentos. Essas representações (modelos) podem ser diagramas, mapas conceituais, modelos matemáticos ou descrições textuais formais, dependendo da área (engenharia, software, dados, negócios etc.).

Segundo FOWLER (2004), UML (*Unified Modeling Language*) é uma família de notações gráficas, apoiada por um meta-modelo único, que ajuda na descrição e no projeto de sistemas de *software*, particularmente daqueles construídos utilizando o estilo orientado a objetos (OO). As linguagens gráficas de modelagem existem há muito tempo na indústria do *software*. O propulsor fundamental por trás de todas elas é que as linguagens de programação não estão em um nível de abstração suficientemente alto para facilitar as discussões sobre projeto. À medida que você trabalha com a UML, a programação fica cada vez mais mecânica, torna-se evidente que esta deve ser automatizada. Na verdade, muitas ferramentas CASE realizam alguma geração de código, o que automatiza a construção de uma parte significativa de um sistema. Finalmente, você chega em um ponto em que todo o sistema pode ser especificado na UML e, assim, chega à UML como linguagem de programação.

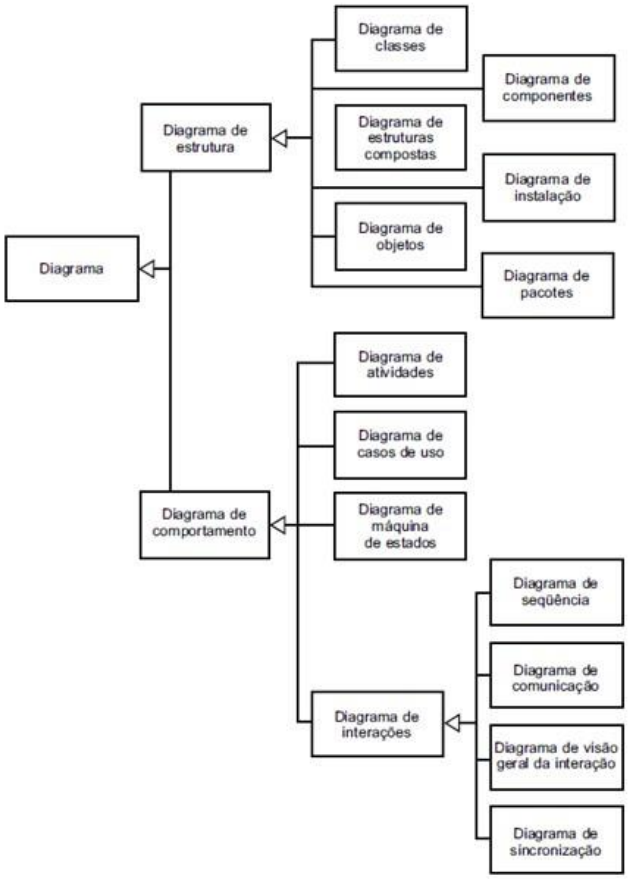


Figura 01: Classificação dos tipos de diagrama UML

Apresentando três modelos clássicos e complementares usados na modelagem de domínio:

ATIVIDADES: CONCEITOS DE MODELOS

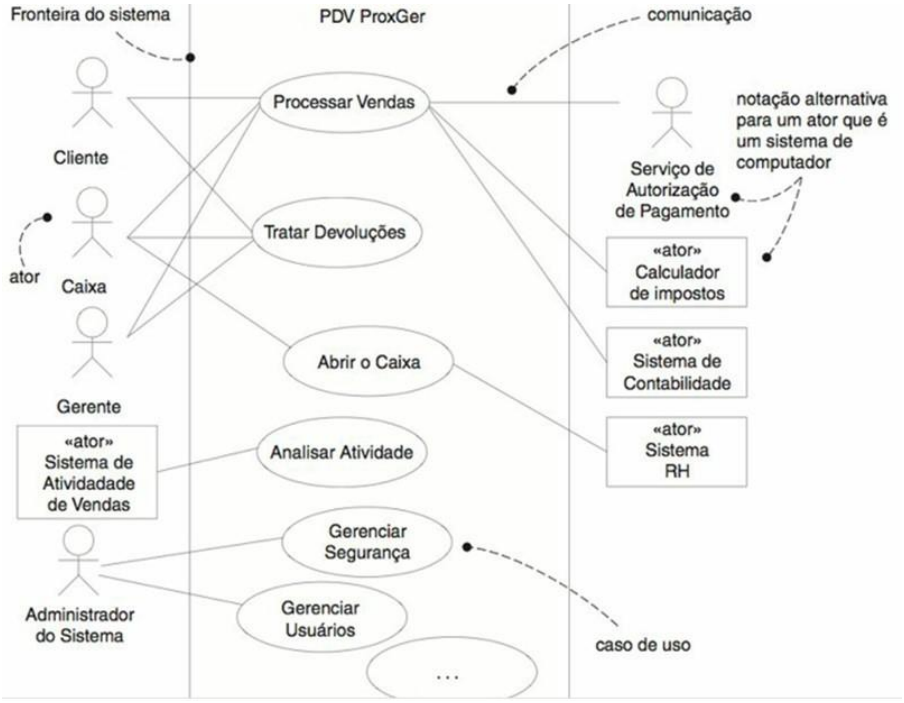
1. Modelo de Casos de Uso (Use Case Model)

Propósito: Representar funcionalidades e interações entre os atores externos (usuários, sistemas) e o sistema em desenvolvimento (LARMAN, 2004)

Características:

- Define o que o sistema faz, não como faz.
- Mostra quem interage e quais são os objetivos dessas interações.
- Ajuda no entendimento dos requisitos funcionais.

Exemplo gráfico:



Esse modelo identifica os atores (Cliente, Caixa, Gerente, Adm do sistema) e as ações (Processar vendas, tratar devoluções, abrir o caixa, gerenciar segurança, gerenciar usuários), representando o escopo funcional do domínio.

2. Modelo de Classes do Domínio (Domain Class Model)

Propósito: Representar as entidades conceituais (objetos do mundo real), seus atributos, relacionamentos e associações. É o modelo estrutural do domínio.

Características:

- Mostra as classes conceituais e seus relacionamentos.
- Permite derivar modelos de dados e arquiteturas orientadas a objetos.
- Usa notação UML – Diagrama de Classes.

ATIVIDADES: CONCEITOS DE MODELOS

Exemplo gráfico:

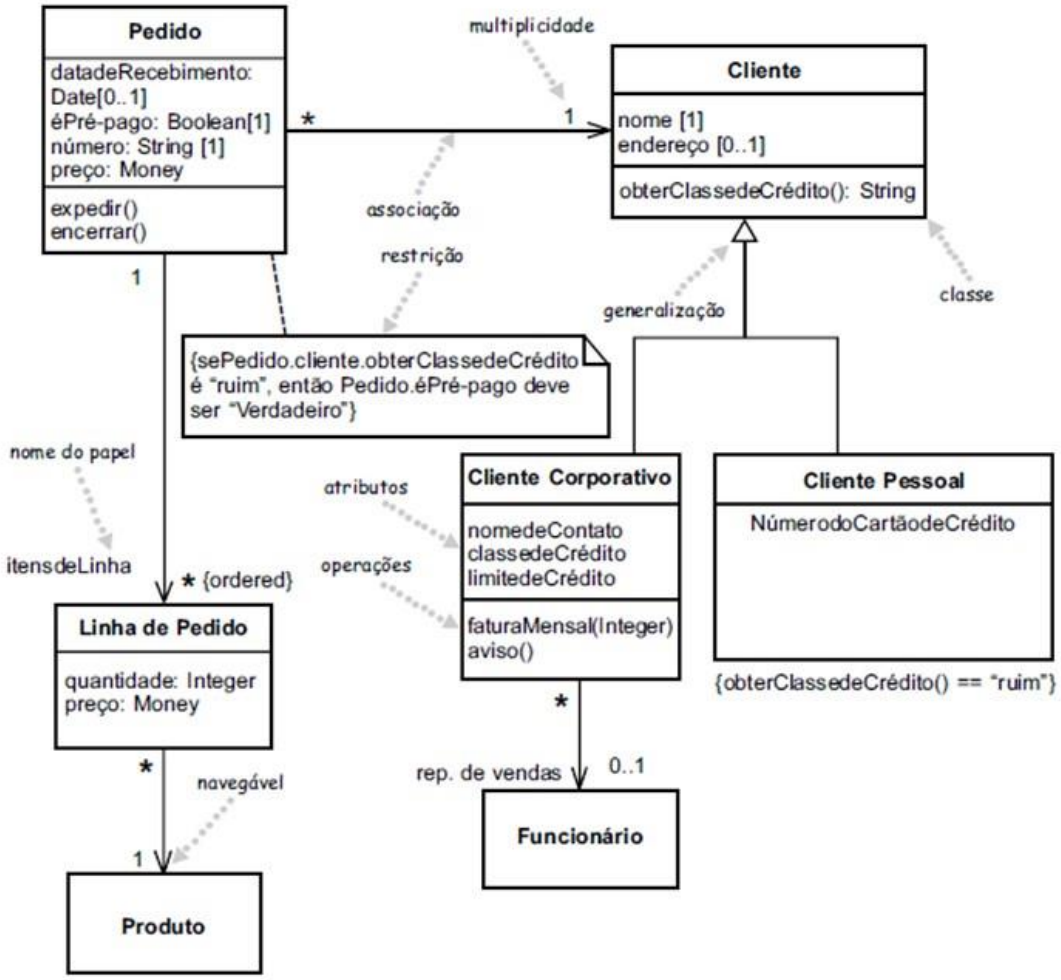


Figura 02: Um diagrama de classe simples

Esse modelo descreve as principais entidades do domínio, mostrando como clientes, pedidos e produtos se relacionam.

3. Modelo de Pacotes e Dependência (Package and Dependency Model)

Propósito: Representar a organização modular do sistema, mostrando como as classes, componentes ou subsistemas estão agrupados em pacotes e como esses pacotes dependem uns dos outros. Seu principal objetivo é visualizar a estrutura hierárquica e as relações de dependência, facilitando o entendimento da arquitetura e da modularização do sistema.

Características:

- Agrupa elementos (classes, componentes, módulos) em pacotes lógicos ou físicos, representando a organização do código.
- Evidencia dependências entre pacotes, isto é, quem utiliza quem.

ATIVIDADES: CONCEITOS DE MODELOS

- Permite analisar acoplamento e impacto de alterações no sistema.
- Auxilia no projeto e manutenção, identificando módulos fortemente ou fracamente acoplados.
- Pode representar tanto estrutura conceitual quanto física (pastas, namespaces, bibliotecas).

Exemplo:

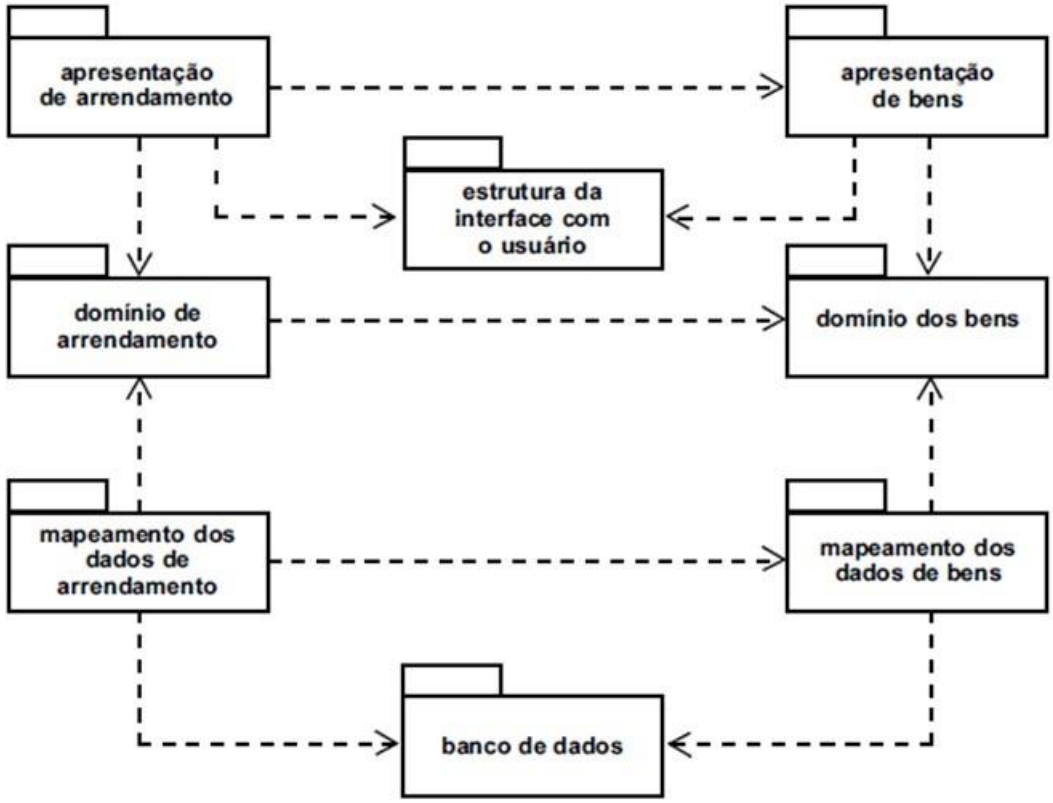


Figura 03: Diagrama de pacotes para uma aplicação comercial

Esse modelo captura uma boa estrutura de pacotes que tem um fluxo claro das dependências.

Resumo

Modelo	Foco Principal	Representação	Pergunta Respondida
Casos de Uso	Funcionalidades do sistema	Diagrama UML	O que o sistema faz e quem o usa?
Classes do Domínio	Estrutura conceitual	Diagrama UML de Classes	Quais entidades existem e como se relacionam?
Casos de Uso Estendidos	Comportamento detalhado	Texto estruturado / BPMN	Como o sistema executa cada função?

ATIVIDADES: CONCEITOS DE MODELOS

4. Conclusão

O uso integrado dos modelos de casos de uso, classes e pacotes constitui uma base sólida para o entendimento, análise e projeto de sistemas orientados a objetos. Cada um desses modelos cumpre um papel complementar dentro do processo de desenvolvimento, permitindo que o sistema seja compreendido de forma gradual do nível mais conceitual até o estrutural.

Assim, a aplicação integrada desses modelos não apenas melhora o entendimento e a comunicação entre analistas, desenvolvedores e clientes, mas também contribui para o desenvolvimento de sistemas mais robustos, coerentes e bem documentados, alinhando o projeto técnico aos objetivos de negócio.

ATIVIDADES: CONCEITOS DE MODELOS

Referências Bibliográficas

1. **BOOCH, G.; RUMBAUGH, J.; JACOBSON, I.** *The Unified Modeling Language User Guide*. 2ª ed. Addison-Wesley, 2005.
2. **FOWLER, Martin.** *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. 3ª ed. Addison-Wesley, 2004.
3. **LARMAN, Craig.** *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*. 3ª ed. Prentice Hall, 2004.