APRENDA UML
POR MEIO DE ESTUDOS DE CASO

Wilson Moraes Góes

Novatec

Editor: Rubens Prates
Revisão gramatical: Marta Almeida de Sá
Editoração eletrônica: Carolina Kuwabata
Capa: Carolina Kuwabata

ISBN: 978-85-7522-346-8

Histórico de impressões:

Janeiro/2014 Primeira edição

Novatec Editora Ltda.
Rua Luís Antônio dos Santos 110
02460-000 – São Paulo, SP – Brasil
Tel.: +55 11 2959-6529
Fax: +55 11 2950-8869
E-mail: novatec@novatec.com.br
Site: novatec.com.br
Twitter: twitter.com/novateceditora
Facebook: facebook.com/novatec
LinkedIn: linkedin.com/in/novatec
VC20131218

CAPÍTULO 1

Introdução à Linguagem Unificada de Modelagem (UML)

A UML é a especificação mais conhecida do Object Management Group (OMG) e é a norma da indústria da informática para descrever graficamente "software". A UML é uma linguagem ou notação visual para especificação (modelagem) de sistemas de informação orientados a objetos. Ela não apresenta um processo fixo para o desenvolvimento de software, ou seja, não é uma metodologia de desenvolvimento de sistemas, é apenas uma notação e pode ser usada com várias metodologias. A UML possui diversos mecanismos de extensão que permitem que ela possa ser utilizada em vários domínios diferentes.

O objetivo da UML é proporcionar aos arquitetos de sistemas, engenheiros de software e desenvolvedores um conjunto de ferramentas para análise, projeto e implementação de sistemas, bem como para modelagem de processos e similares.

1.1 Por que usar a UML

Ela pode ser utilizada no apoio às fases de análise, projeto e implementação, pois nos permite pensar antes de codificar. Trabalha com um conjunto de diagramas que permitem uma notação clara e consistente. Facilita a comunicação entre a equipe de desenvolvimento, bem como dela com os usuários do sistema, aumentando assim a participação do time do projeto. Ajuda a apontar e/ou prever inconsistências e omissões. Pode ser usada em projetos de todos os tamanhos. Não se prende a nenhuma metodologia. Trata dados e processos de maneira integrada. Ajuda a conceber nossas ideias, em relação ao sistema que estivermos projetando. Por fim, possibilita documentar os artefatos do sistema.

1.2 Breve histórico

Nos anos 80 e 90, os desenvolvedores tinham à disposição um conjunto de métodos e notações com iniciativas de desenvolvimento isoladas, simbologias, processos e ferramentas distintas e que afetivamente não contribuíam com a melhoria expressiva no processo de desenvolvimento de software.

Porém a partir da união de um sueco – Ivar Jacobson, cientista da computação – e de dois norte-americanos – Grady Booch e James Rumbaugh, respectivamente engenheiro de software e cientista da computação – é que teve início a proposta de criação de um modelo único que veio a ser a UML, lançada em 1996 (versão 0.9); originada dos três principais métodos orientados a objetos (Booch, OMT e OOSE), incorporou uma série de melhores práticas de construção de sistemas orientados a objetos.

1.3 Diagramas da UML

Na versão 2.0, a UML possui treze tipos de diagrama, divididos em três categorias: seis representam a estrutura de aplicação estática, três representam tipos gerais de comportamento e quatro representam diferentes aspectos das interações:

- **Diagramas de estrutura** incluem o Diagrama de classes, Diagrama de objetos, Diagrama de componentes, Diagrama de estrutura composta, Diagrama de pacotes e Diagrama de implantação.
- **Diagramas de comportamento** incluem o Diagrama de casos de uso, Diagrama de atividades e Diagrama de máquina de estado.
- **Diagramas de interação**, todos os derivados do diagrama de comportamento mais geral, incluem o Diagrama de sequência, Diagrama de comunicação, Diagrama de tempo e Diagrama de visão geral da interação.

Neste livro não vamos abordar os diagramas de estrutura composta, tempo e visão geral, pois no dia a dia das equipes de desenvolvimento de software eles não são utilizados.

1.3.1 Diagrama de Casos de Uso

A principal função deste diagrama é apresentar um sistema de informação pela visão do usuário, ou seja, quais são os módulos que compõem o sistema, quem são seus usuários e quais papéis cada um vai desempenhar para seu funcionamento (Figura 1.1).

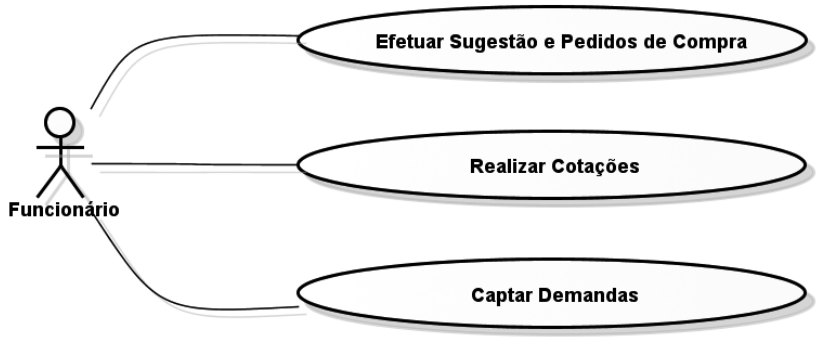

Figura 1.1 – Exemplo de Diagrama de Caso de Uso.

1.3.2 Diagrama de Atividade

Este diagrama representa os aspectos dinâmicos e pode ser utilizado para modelar um sistema de informação, alguns módulos desse sistema, uma pequena parte do código de um de seus programas, um algoritmo ou os processos (fluxos de trabalho) de uma organização (Figura 1.2).

Figura 1.2 – Exemplo de Diagrama de Atividade.

1.3.3 Diagrama de Classes

Este diagrama enfatiza os dados que serão necessários para a construção do sistema de informação. Sua ideia central é concentrar a construção de um sistema em torno de objetos, ou seja, mais próximo do mundo real. Representa os dados de maneira estática (Figura 1.3).

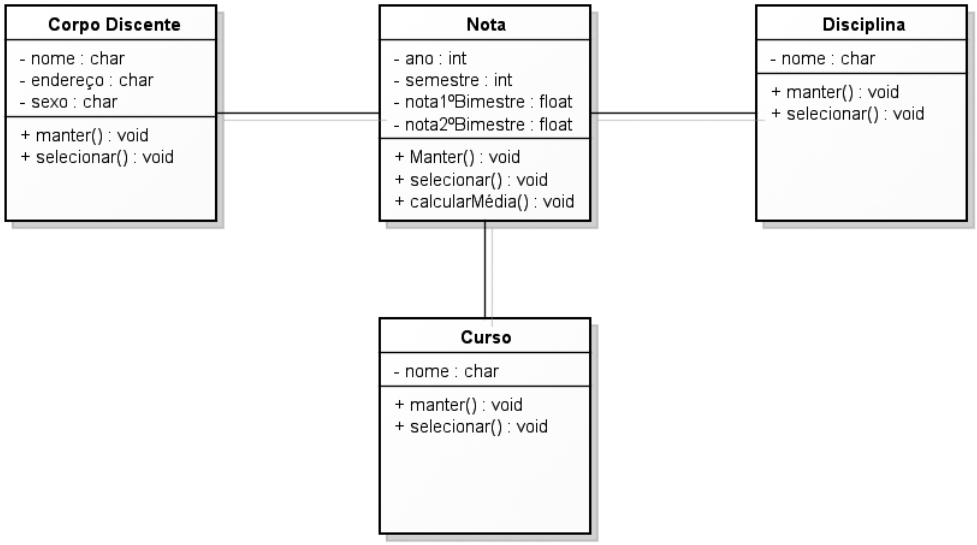

Figura 1.3 – Exemplo de Diagrama de Classes.

1.3.4 Diagrama de Objetos

Um diagrama de objetos representa uma instância de uma classe específica, ou seja, mostra os valores armazenados pelos objetos das classes num determinado intervalo de tempo. Ele também é estático, logo, havendo a necessidade de mostrar o objeto em outro momento de sua existência, uma nova visão do diagrama deve ser desenvolvida. Ele também mostra um conjunto de objetos e seus relacionamentos e também pode conter elementos como: notas, restrições e pacotes (Figura 1.4).

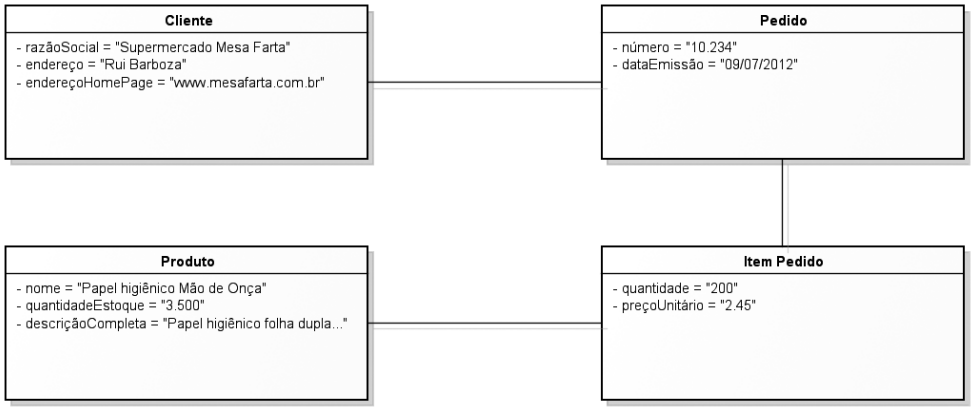

Figura 1.4 – Exemplo de Diagrama de Objetos.

1.3.5 Diagrama de Pacotes

Serve para modelagem estrutural do sistema numa visão de alto nível dividindo-o em partes lógicas e descrevendo as relações entre elas (Figura 1.5).

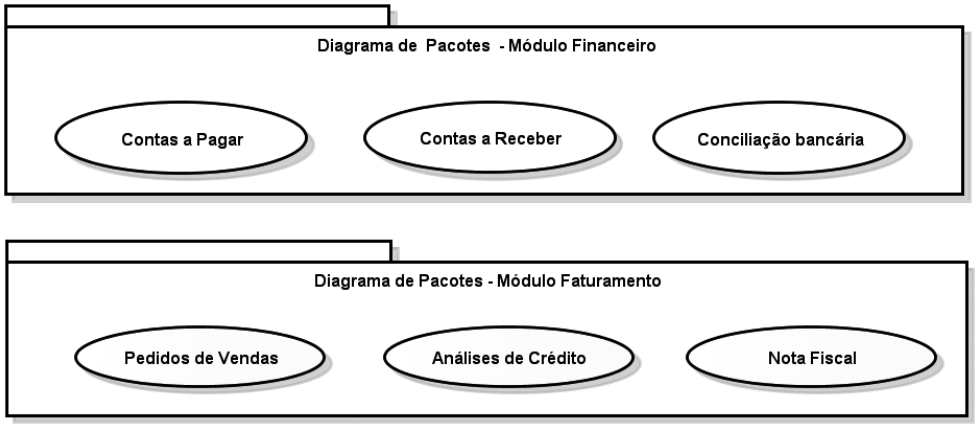

Figura 1.5 – Exemplo de Diagrama de Pacotes.

1.3.6 Diagrama de Comunicação

Ilustra a colaboração dinâmica entre os objetos, ou melhor, o fluxo das mensagens entre um objeto e outro. É bem parecido com o diagrama de sequência, porém não possui o aspecto temporal e concentra-se em como os objetos estão vinculados por meio de mensagens que são numeradas para indicar a sequência (Figura 1.6).

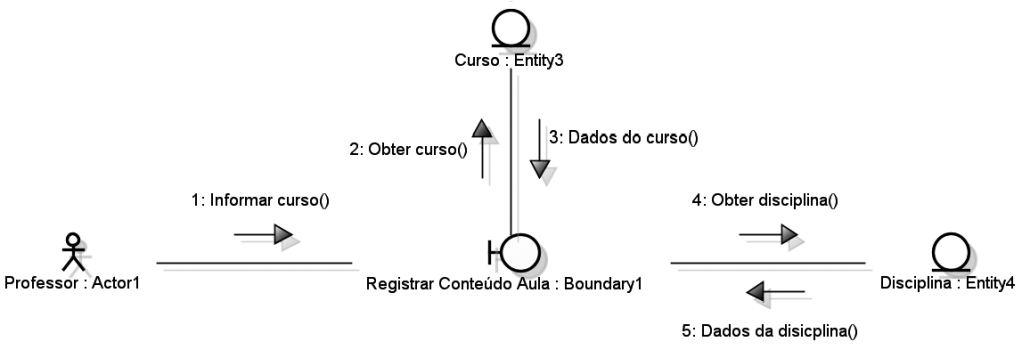

Figura 1.6 – Exemplo de Diagrama de Comunicação.

1.3.7 Diagrama de Máquina de Estado

Este tipo de diagrama é utilizado para capturar o ciclo de vida das classes, porém também pode ser utilizado para modelar o comportamento de casos de usos, sistemas ou subsistemas (Figura 1.7).

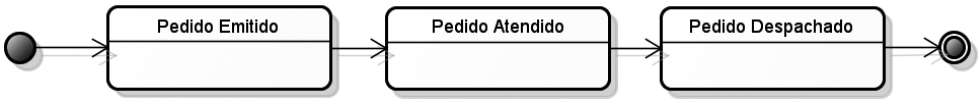

Figura 1.7 – Exemplo de Máquina de Estado.

1.3.8 Diagrama de Sequência

Diagramas de sequência descrevem ao longo de uma linha de tempo a sequência de comunicações entre objetos de um sistema de informação. Seus principais objetivos são: documentar casos de uso, mostrar como os objetos do sistema se comunicam por meio de mensagens em ordenação temporal, validar se todas as operações das classes foram identificadas e declaradas ou ainda validar a existência de um objeto necessário ao funcionamento do sistema (Figura 1.8).

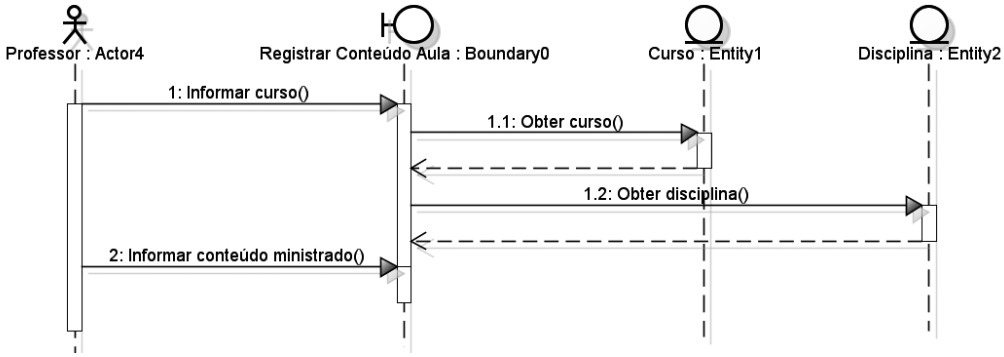

Figura 1.8 – Exemplo de Diagrama de Sequência.

1.3.9 Diagrama de Implantação

Consiste na organização do conjunto de elementos (hardware e software) de um sistema para a sua execução. É útil em projetos em que é preciso representar a estrutura de software (sistema gerenciador de banco de dados, sistema operacional) e hardware (servidores, estações de trabalho, switch, conexões de rede etc.) onde o sistema de informação será executado (Figura 1.9).

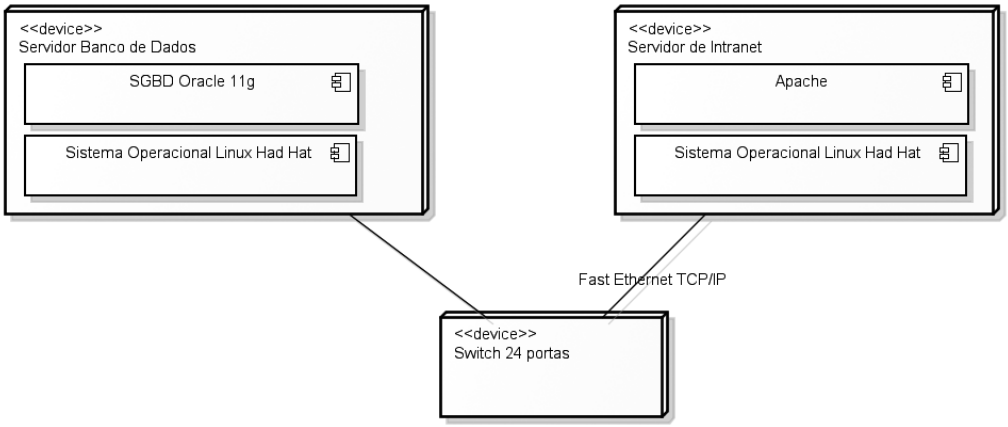

Figura 1.9 – Exemplo de Diagrama de Implantação.

1.3.10 Diagrama de Componentes

É utilizado para modelar e documentar como estão estruturados os arquivos físicos de um sistema, permitindo assim uma melhor compreensão e facilitando a reutilização de artefatos (Figura 1.10).

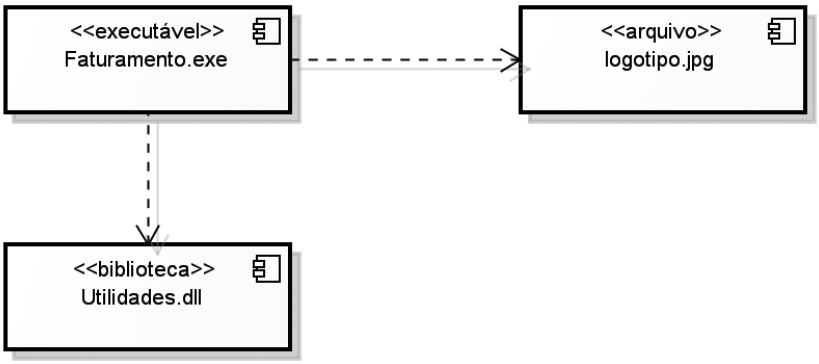

Figura 1.10 – Exemplo de Diagrama de Componentes.