INSTITUTO FEDERAL DE MATO GROSSO DO SUL
CAMPUS NAVIRAÍ

NOTAS DE AULA DE

ANÁLISE E PROJETO ORIENTADO A OBJETO II

Prof. MSc. Luiz F. Picolo

NAVIRAÍ - MS

Atualizado em 27 de agosto de 2019

2 Diagrama de classes

Dedique-se aos estudos para que eles o façam
melhor para a sociedade e para si mesmo.
Alvaro Granha Loregian

O modelo de caso de uso é formado por diversos elementos que tem como propósito o fornecimento de uma visão do ponto de vista externo ao sistema. Essa funcionalidade, externamente observada, é fornecida por meio da colaboração entre diversos elementos, os quais são chamados de **objetos**. A colaboração entre os objetos podem ser vista em dois aspectos, os **dinâmicos** e **os estruturais estáticos**.

O aspecto dinâmico diz respeito a forma com que os objetos realizam as trocas de mensagens e as reações a eventos que ocorrem no sistemas. Já os estruturais estáticos permite visualizar como o sistema está estruturado internamente para que as funcionalidades visíveis sem produzidas [Bezerra 2016].

Assim, realizaremos uma introdução aos aspectos estruturais estáticos, demostrando os elementos que o diagrama de classes utiliza para a representar um sistema orientado a objeto. O diagrama de classe, segundo Guedes 2018, é um dos mais importante e mais utilizados diagramas da UML, o qual permite a visualização da **classes** que comporão o sistemas com seus respectivos **atributos**, **métodos** e pelo relacionamento entre estes.

2.1 Modelo de classes

A medida que o sistemas é desenvolvido, o modelo de classe pode evoluir para demostrar o sistema com maior riqueza de detalhes. Assim, podemos estudar três níveis de detalhamento: **domínio**, **especificação** é **implementação** [Bezerra 2016].

- **Modelo de domínio:** Nesse modelo não é levado em consideração como o sistemas será implementado mas apenas as classes que o comporão.
- **Modelo de especificação:** Já no modelo de especificação são adicionados detalhes específicos relacionados a solução do software.
- **Modelo de implementação:** Esse modelo corresponde a implementação das classes em alguma linguagem de programação.

2.2 Diagrama de classes

O diagrama de classes, como dito anteriormente, é um dos mais importantes para o desenvolvimento de um sistema orientado a objeto.

Como todos os diagramas da UML, o diagrama de classe é composto de elementos que, consequentemente, possuem também suas responsabilidades. Assim, veremos todos este elementos, essenciais para o modelo de domíno, e como eles interagem entre si.

Lembre-se que, o conceito relacionado a cada um já foi apresentado no Capítulo **??**.

2.2.1 Classes, atributos e métodos

Uma classe é representada por meio de uma **caixa** com, no máximo, três compartimentos. No compartimento superior é exibido o nome da classe, o qual é o único elemento obrigatório.

No segundo compartimento são declarados os atributos. Este, como visto no **??**, armazenam os dados que um objeto armazena. Por fim, no último compartimento são declarados os métodos.

Assim, segundo Bezerra 2016, as possíveis anotações para que possamos representar uma classe utilizando as notações UML são representadas na Figura 1.

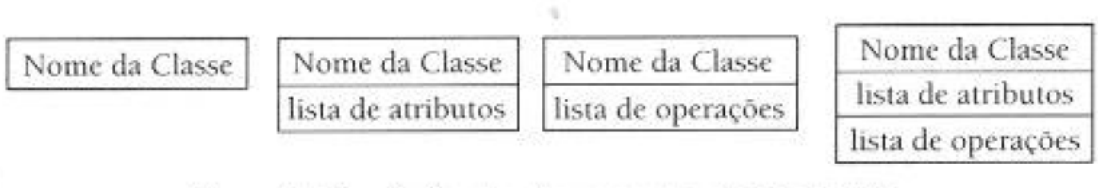

Figura 1 – Possíveis notações para uma classe em UML.

Fonte: [Bezerra 2016]

Por convenção o nome das classe sempre será no singular e iniciará com letra maiúscula usando sempre o a forma de escrita conhecida como *CamenCase*. Já os atributos e métodos sempre iniciarão com letra minúscula, mas também seguirão o padrão *CamenCase*. Os métodos devem expressar uma ação que o objeto da classe sabe realizar. Vejamos um exemplo da classe ***ContaBancaria***.

Figura 2 – Diferentes graus de detalhe para *ContaBancaria*.

Fonte: [Bezerra 2016]

2.3 Exercícios de fixação

Utilizando às notações aprendidas, identifique as possíveis classes, atributos e métodos para o problema abaixo.

1. O professor **Picolo** precisa de um software para lhe ajudar a lembrar os aniversários de seus alunos. O software deve ser capaz de armazenar o nome do aluno, sua data de nascimento, seu e-mail, seu sexo e qualquer outro dado relevante. O software deve permitir o cadastro dos alunos, avisar quem faz aniversário no dia, calcular e mostrar a idade do aluno, exibir um relatório de aniversariantes no mês e outras funcionalidades que se julgar necessário.

2. Faça o mesmo para o exercício 1 do exercício de fixação **??**

2.3.1 Relacionamentos

As classes costumam ter relacionamentos entre si, com o objetivo de compartilhar informações e colaborar uma com as outras para permitir a execução dos diversos processos executados em um sistema [Guedes 2018]. Assim, vermos as diversas formas de relacionamentos presentes na UML e relacionadas ao diagrama de classes.

2.3.1.1 Associação

Uma associação descreve um vínculo que ocorre normalmente entre duas classes. Esse ligação da-se o nome de **associação binária**. Contudo, há casos que a classe pode se associar a si mesmo, dando origem a um tipo de associação **chamada de unária**, ou associar-se a outras classes, a qual é chamada de **associação N-ária**. As associa-

ções representam a maneira como as classes estão unidas e interagem para realizar o compartilhamento de informações.

Muito similar ao diagrama de Casos de Uso, as associações em um diagrama de classes é representada por um linha sólida ligando as classes envolvidas, podendo também conter setas que representam a **navegabilidade**, ou seja, o sentido que as informações são transmitidas.

As associações também podem conter **títulos** que, juntamente com a navegabilidade, ajudam a compreender melhor o fluxo das informações e a forma com que ela será transmitida de um objeto para outro.

2.3.1.2 Associação unária

Este tipo de associação acontece quando há o relacionamento de uma classe consigo mesma.

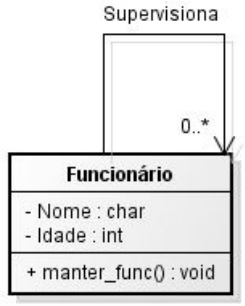

Figura 3 – Associação unária

Fonte: http://sgvclin.altervista.org/rea-uml/pop/pop-3/popup-texto3.html

Percebam que existe uma associação e próximo ao final da seta temos **0..***. Esta informação representa a multiplicidade. A **multiplicidade** é extremamente semelhante a **cadinalidade** utilizada no Modelo Entidade-Relacionadal. Ela procura determinar qual das classes envolvidas em uma associação fornece informações para as outras, além de especificar o nível de dependências.

0..1	Zero ou um.
1..1	Um e somente um.
0..*	Zero ou muitos.
*	Muitos.
1..*	No mínimo um ou muitos.
3..5	Mínimo de três e máximo de cinco.

Figura 4 – Multiplicidades

Fonte: http://sgvclin.altervista.org/rea-uml/pop/pop-3/popup-texto3.html

2.3.1.3 Associação binária

Uma associação binária ocorre quando são identificados relacionamentos entre duas classes. Este tipo de associação é o mais comum no diagrama de classe.

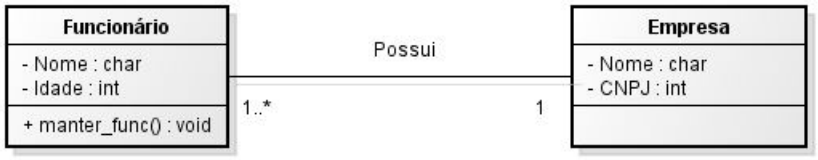

Figura 5 – Associação Binária

Fonte: http://sgvclin.altervista.org/rea-uml/pop/pop-3/popup-texto3.html

2.3.1.4 Associação N-ária

Já, uma associação N-ária, também conhecida como ternária, são aquelas mais de duas classes. Neste, além da linha associando as classes, também possuímos um losango para onde converge todas as ligações.

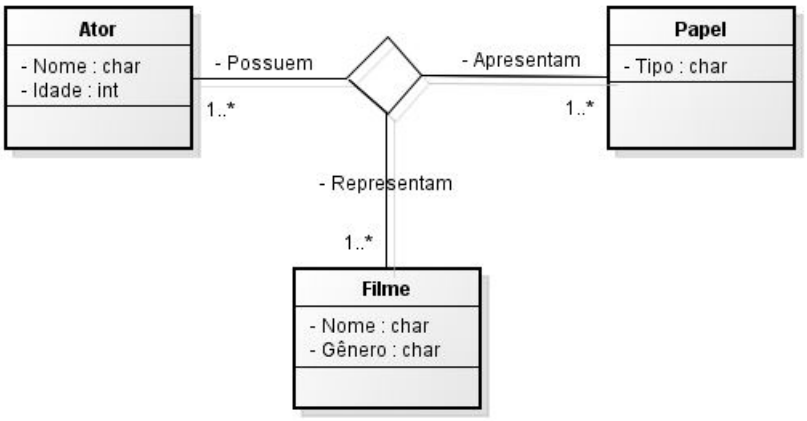

Figura 6 – Associação N-ária

Fonte: http://sgvclin.altervista.org/rea-uml/pop/pop-3/popup-texto3.html

2.3.1.5 Agregação

É um tipo especial de associação em que tenta-se demonstrar que as informações de um objeto (chamado objeto-todo) precisam ser complementados pelas informações contidas em um ou mais objetos de outra classe (chamados objetos-parte); conhecemos como todo/parte.

Esse tipo de associação é representada por um losango sem preenchimento no final da associação, sempre do lado do objeto **todo**.

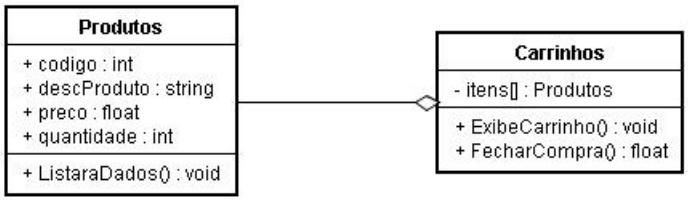

Figura 7 – Agregação

Fonte: http://www.cpscetec.com.br/adistancia/aula5.html

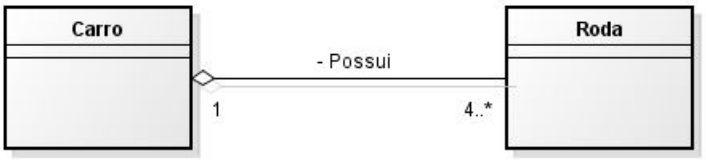

Figura 8 – Agregação

Fonte: http://sgvclin.altervista.org/rea-uml/pop/pop-3/popup-texto3.html

2.3.1.6 Composição

Pode-se dizer que composição é uma variação da agregação. Uma composição tenta representar também uma relação todo - parte. No entanto, na composição o objeto-pai (todo) é responsável por criar e destruir suas partes. Em uma composição um mesmo objeto-parte não pode se associar a mais de um objeto-pai.

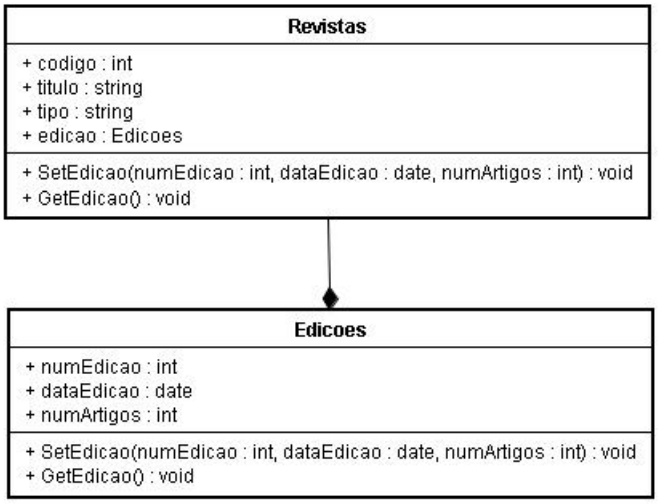

Figura 9 – Composição

Fonte: http://www.cpscetec.com.br/adistancia/aula5.html

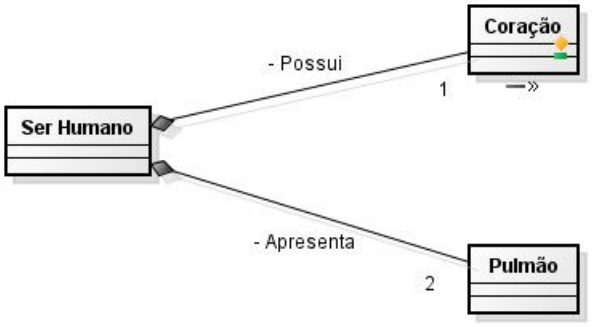

Figura 10 – Composição

Fonte: http://sgvclin.altervista.org/rea-uml/pop/pop-3/popup-texto3.html

2.3.1.7 Especialização/Generalização

Especialização/Generalização ou também chamada de Herança. Este tipo de associação ocorre quando uma classe gera novas classes que sejam suas cópias perfeitas e a partir destas classes é possível adaptá-las ao meio em que sejam necessárias. A classe que é usada como referência é conhecida por **superclasse / generalização** ou classe mãe, a classe criada com base nessa superclasse é conhecida como **sub-classe / especialização** ou classe filha. Na classe filha, podemos redefinir métodos e criar novos campos.

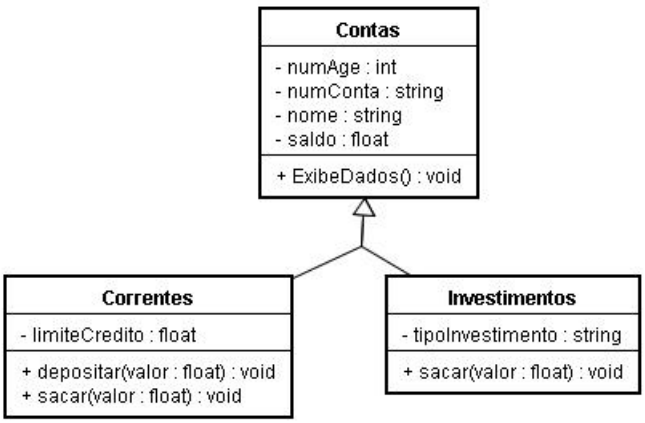

Figura 11 – Especialização/Generalização

Fonte: http://www.cpscetec.com.br/adistancia/aula5.html

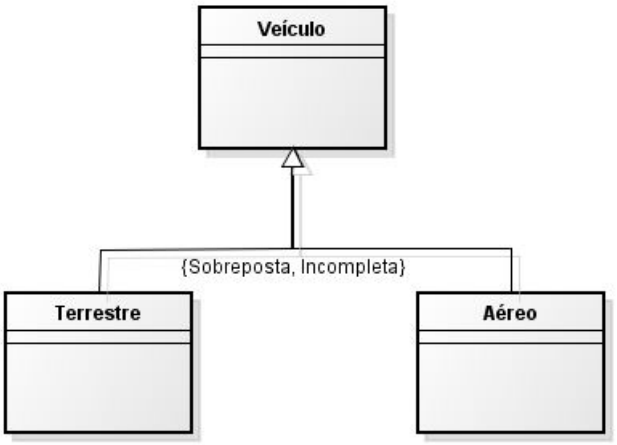

Figura 12 – Especialização/Generalização

Fonte: http://sgvclin.altervista.org/rea-uml/pop/pop-3/popup-texto3.html

2.3.2 Encapsulamento

Conceitua-se encapsulamento como sendo o processo utilizado para proteger os campos e operações de uma classe (atributos e métodos), permitindo que apenas os membros públicos sejam acessados pelos usuários de determinada classe.

Para atingir o encapsulamento, uma das formas é definindo a visibilidade das propriedades e dos métodos de um objeto. A visibilidade define a forma como essas propriedades devem ser acessadas.

Público - public	Nível de acesso livre. Indica que o método ou atributo da classe é público, ou seja, pode-se acessar este atributo, ou executar esse método, por qualquer código de programa. Usamos o sinal (+) na notação UML para simbolizar essa visibilidade.
Privado - private	Nível mais protegido. Mebros declarados como private, só podem ser acessados dentro da classe em que foram declarados. Usamos o sinal (-) na notação UML para simbolizar essa visibilidade.
Protegido - protected	Nível de acesso intermediário. Serve para que o método ou o atributo seja público dentro do código da própria classe e de qualquer classe que herde daquela onde está o método ou propriedade protected. É privado e não acessível de qualquer outra parte. Usamos o sinal (#) na notação UML para simbolizar essa visibilidade.

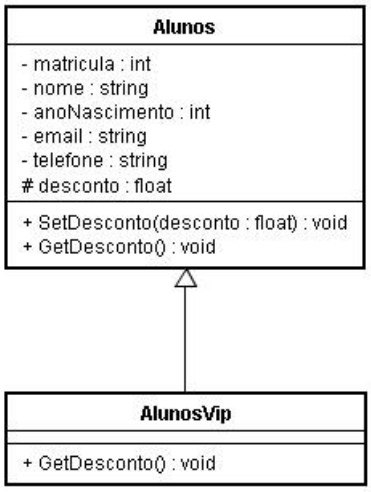

Figura 13 – Encapsulamento

Fonte: http://www.cpscetec.com.br/adistancia/aula5.html

2.3.3 Exercícios de fixação

1. Uma empresa possui fornecedores de peças. Aos fornecedores é atribuído um código que os identifica univocamente. O mesmo procedimento é usado para as peças. Os fornecedores possuem ainda um nome, endereço e telefone. As peças são caracterizadas por designação e cor. As cores podem ser preto, branco, cinzento e vermelho. Os fornecimentos são feitos a um dado preço e numa dada data.

2. Cada escola da comunidade Alfa é dividida em um ou mais departamentos (letras, matemática, etc.). Um departamento é chefiado por um de seus professores, mas há casos em que esse cargo está vago. Não há acúmulo de chefia. Professores podem estar alocados em um ou mais departamentos. Um departamento pode ser criado sem que haja professores a ele alocados. Um aluno pode estar matriculado em mais de uma escola e pode frequentar mais de um curso na mesma escola. Escolas podem não ter alunos matriculados. Cada departamento tem seu conjunto específico de cursos (pelo menos um). Cada curso pode ser ministrado por um ou mais professores. Cada professor pode ministrar qualquer número de cursos.

3. Para cada obra armazena-se o ISBN, o(s) autor(es), o título, a editora, o assunto e a edição. Os exemplares possuem a data de inclusão no acervo e uma marcação de disponibilidade (as bibliotecas só cadastram obras que constam do acervo). As bibliotecas emprestam livros aos alunos e aos membros da comunidade. Alunos podem retirar até 2 títulos; membros da comunidade podem retirar apenas um título por vez.

4. Uma empresa efetua encomendas, numeradas. Cada encomenda possui um número variável de itens, cada um das quais diz respeito a um produto a ser encomendado. Os itens possuem uma ordem numérica dentro de cada encomenda. A encomenda tem uma data e dirige-se a um só fornecedor. Tanto os produtos como os fornecedores são identificados por um código próprio. Os itens incluem o número dom item, o código do produto, o preço unitário e a quantidade encomendada.

Referências

BEZERRA, E. *Princípios de Análise e Projeto de Sistema com UML*. [S.l.]: Elsevier Brasil, 2016. v. 3. Citado 3 vezes nas páginas 2, 3 e 4.

GUEDES, G. T. *UML 2-Uma abordagem prática*. [S.l.]: Novatec Editora, 2018. Citado 2 vezes nas páginas 2 e 4.

PRESSMAN, R.; MAXIM, B. *Engenharia de Software-8ª Edição*. [S.l.]: McGraw Hill Brasil, 2016. Nenhuma citação no texto.

RUIZ, E. E. S. *IBm1030 Programação Orientada a Objetos*. 2008. Disponível em: <http://dcm.ffclrp.usp.br/~evandro/ibm1030/constru/heranca.html>. Nenhuma citação no texto.

SILVA, D. Lucas da et al. Ontologias e unified modeling language: uma abordagem para representação de domínios de conhecimento. v. 10, 10 2009. Nenhuma citação no texto.

SOMMERVILLE, I. *Engenharia de Software. 6ª*. [S.l.: s.n.], 2003. Nenhuma citação no texto.