EXEMPLO PRÁTICO AULA 3 - Herança, Agregação e Associação

Consideremos que uma faculdade deseje desenvolver um sistema para controlar

as orientações e defesas dos Trabalhos de Conclusão de Curso (TCC). Para isso, utilizará a Orientação a Objetos para identificar os principais conceitos que, posteriormente, serão implementados usando uma linguagem orientada a objetos.

Destacamos as seguintes funcionalidades referentes ao sistema de controle de orientações de TCC:

a) armazenamento de dados de alunos, professores e cursos da instituição;

b) identificação dos professores que trabalham como coordenadores de curso;

c) registro das matrículas dos alunos nos cursos;

d) armazenamento dos dados sobre as orientações de TCC;

e) cadastramento das informações sobre as bancas de defesa de TCC realizadas na instituição.

Com base nas funcionalidades descritas, iniciamos o processo de abstração para identificar os

objetos envolvidos no novo sistema. Após uma breve análise, identificamos os seguintes objetos:

a) aluno;

b) professor;

c) curso;

d) coordenador;

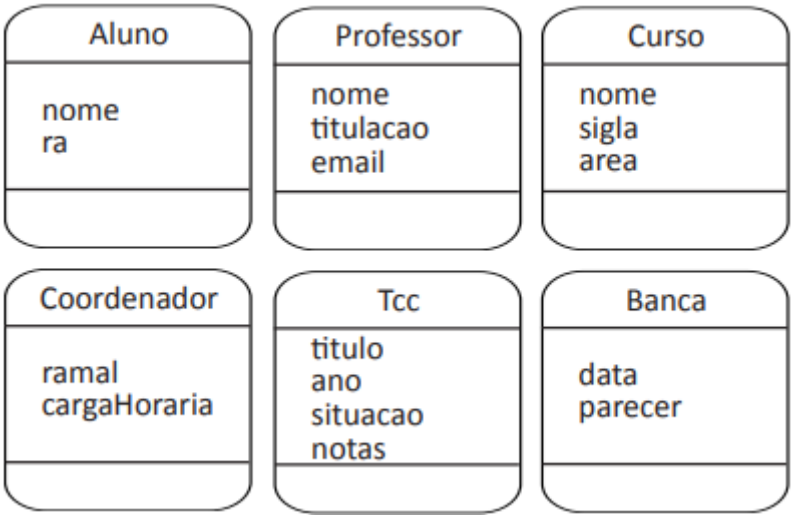
e) TCC;

f) banca.

Muitas dúvidas devem surgir no momento da abstração dos objetos, pois essa é uma dificuldade comum para aqueles que estão iniciando o aprendizado dos conceitos da Orientação a Objetos. Note que os objetos sempre possuirão informações que os descrevem (por exemplo, o objeto Aluno possui as seguintes características: nome, ra, telefone, endereço etc.).

Assim, no momento da abstração e da identificação dos objetos, você pode se orientar procurando informações que deverão de algum modo aparecer em seu sistema. Ao analisar quem é o “dono” da informação, você descobre o objeto que possui aquela característica.

Sabendo que objetos podem ser representados por uma classe que reúne em sua estrutura as características e os comportamentos comuns a eles, detalhamos, a seguir, as classes inicialmente identificadas, por meio de um diagrama de classes da representação UML (Figura abaixo).



Note que nesse momento representamos apenas as características como atributos nas classes,

enquanto os comportamentos serão identificados posteriormente, quando estivermos analisando as relações existentes entre as classes. Lembre-se de que os nomes das classes, seus atributos e métodos devem ser identificados sem o uso de acentuação e símbolos especiais, pois posteriormente serão codificados usando uma linguagem de programação. Nas linguagens de programação, incluindo Java, o uso de acentuação e de símbolos especiais não é permitido.

Uma vez que foram identificadas as classes, vamos agora analisar as relações entre elas.

Destacamos as seguintes:

- Herança.
- Agregação/Composição.
- Associação.

Observamos que a Herança é a relação que permite definir uma nova classe usando como base as definições já existentes em outra classe. A pergunta-chave que podemos fazer para auxiliar na identificação da herança entre classes é se um objeto da classe A “é um” objeto da classe B. Assim, ao analisarmos as classes identificadas na Figura 13, notamos que existe uma relação de herança entre as classes Professor e Coordenador. Isto é, um Coordenador “é um” Professor.

Expandindo um pouco mais essa análise, constatamos que todo Coordenador “é um” Professor, mas nem todo Professor “é um” Coordenador. Por meio da herança entre as classes Professor e Coordenador, temos que a superclasse Professor estende suas definições para a subclasse Coordenador. A classe Coordenador herda todas as características existentes na classe Professor, o que possibilita o reaproveitamento de código e a facilidade de manutenção de código.

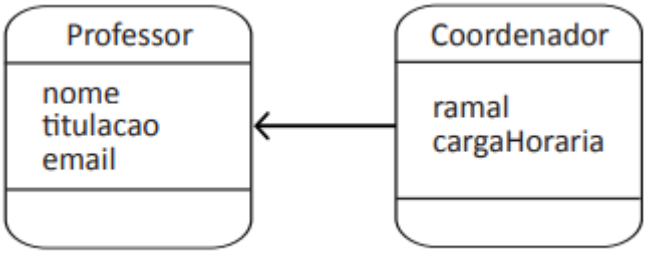


Figura 14 *Herança entre as Classes Professor e Coordenador (UML).*

Analisemos agora a relação de Agregação/Composição. Esta relação possibilita utilizar como parte da definição de uma classe outra(s) classe(s) já existente(s). A maior motivação de seu uso é permitir a reutilização de classes. Em nosso sistema, podemos identificar a relação de agregação entre a classe Banca e a classe Professor (Figura 15), pois os Objetos Banca e Professor podem existir independentemente uns dos outros. A Composição não é definida, nesse caso, porque os objetos podem existir separados uns dos outros.

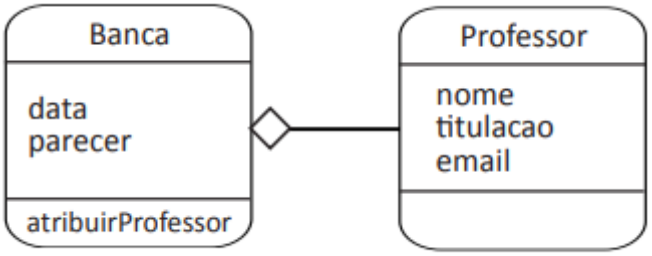


Figura 15 *Agregação entre as Classes Banca e Professor (UML).*

A Agregação entre Banca e Professor pode ser expressa pela indicação de que uma Banca possui como parte de sua definição a participação de dois Professores. A relação parte-todo caracteriza a agregação entre classes. Professor é parte da definição de uma Banca. Note que existe a indicação, na seta de agregação, de que dois Professores participam da agregação com a classe Banca (Figura 15).

Como reflexo da agregação entre classes, novos comportamentos podem ser adicionados às classes existentes. Assim, adicionamos o método "atribuirProfessor" à classe Banca, que será responsável por criar o vínculo entre as classes Banca e Professor (Figura 15). A quantidade de comportamentos adicionados às classes depende única e exclusivamente da análise que se realiza. É importante ressaltar que uma vez adicionado um comportamento a uma classe, este deverá ser codificado posteriormente no momento da programação orientada a objetos.

Outra agregação que podemos destacar é a que ocorre entre Curso e Coordenador e que um Coordenador está relacionado a um Curso. Assim, a classe agregadora Curso possui como parte de sua definição o Coordenador que o coordena. Note que, quando apenas um objeto participa da relação, não precisamos identificar a quantidade junto à seta de agregação do diagrama de classes da UML.

Como reflexo da agregação entre Curso e Coordenador, adicionamos um novo comportamento à classe Curso chamado: "trocarCoordenador". Este será representado no diagrama de classes (Figura 16) em forma de um método, quando a classe Curso for codificada usando uma linguagem orientada a objetos.

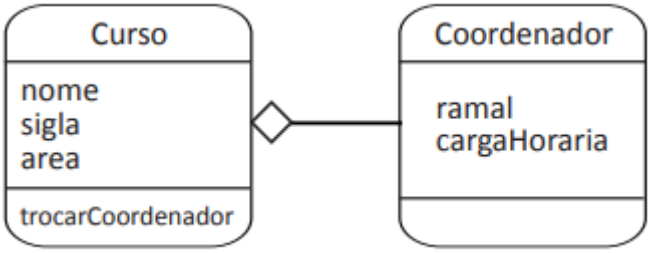


Figura 16 *Agregação entre as Classes Curso e Coordenador (UML).*

Outra agregação existente no sistema relaciona a classe Tcc com a classe Aluno, indicando que um Tcc utiliza como parte de sua definição o aluno responsável pelo trabalho. Como forma de permitir que um Aluno seja vinculado ao Tcc, adicionamos o comportamento "vincularAluno" na classe Tcc (Figura 17).

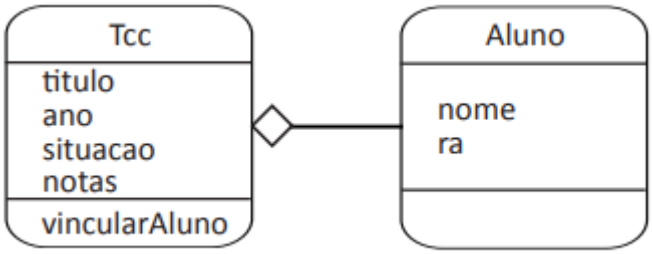


Figura 17 *Agregação entre as Classes Tcc e Aluno (UML).*

A composição é outra relação existente no sistema que relaciona a classe Tcc com a classe Banca, indicando que um Tcc utiliza como parte de sua definição a banca que examinará o trabalho. Na Composição, um objeto do tipo Banca é relacionado ao objeto do tipo Tcc, e o objeto Banca só existirá se estiver vinculado ao objeto Tcc. Quando o objeto Tcc deixar de existir, o objeto Banca também será destruído. Como forma de permitir que uma Banca seja vinculada ao Tcc, adicionamos o comportamento "vincularBanca" na classe Tcc (Figura 18)

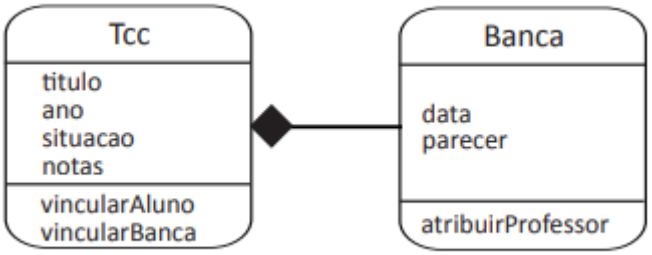


Figura 18 *Agregação entre as Classes Tcc e Banca (UML).*

Por fim, analisaremos a relação de Associação. A associação permite representar as relações entre objetos que apresentam estruturas e comportamentos diferenciados, e não se caracterizam por vínculos do tipo "parte-todo" (agregação ou Composição) e "é um" (herança). Como exemplo, destaca-se a relação entre as classes Aluno e Curso, em que um aluno se matricula em um determinado Curso.

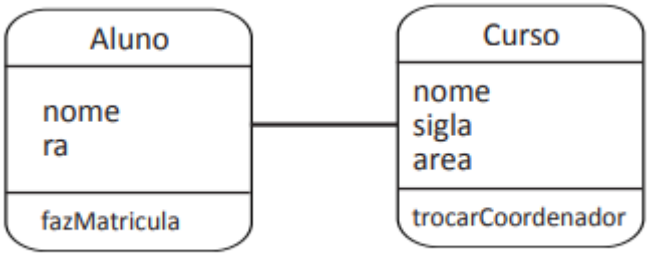
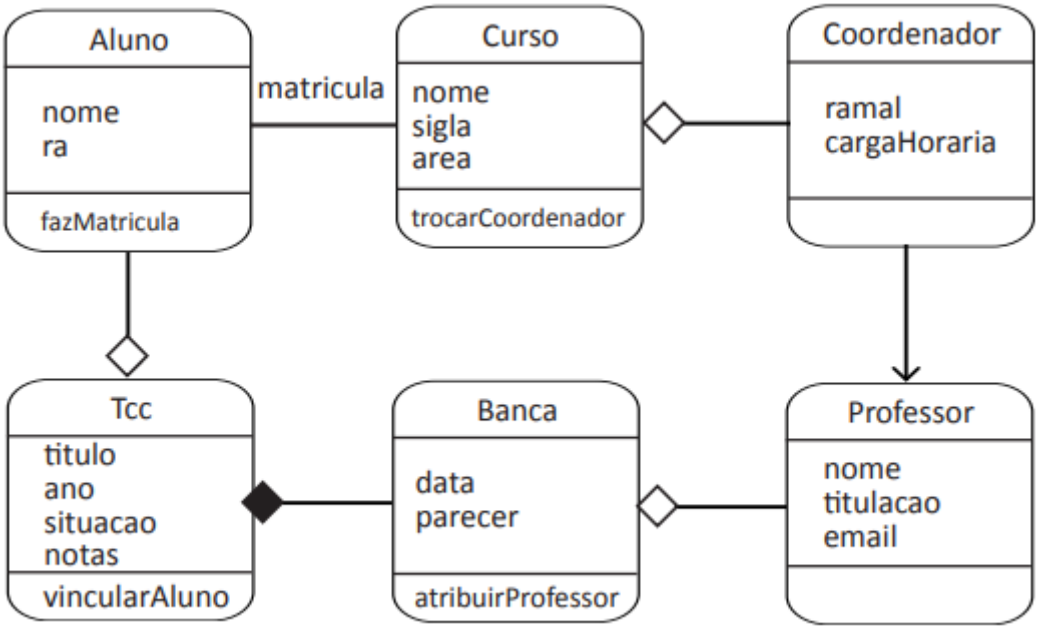


Figura 19 *Associação entre as Classes Aluno e Curso (UML).*

A associação matrícula pode resultar na criação de comportamentos nas duas classes relacionadas. Note que podemos criar um novo comportamento na classe Aluno chamado "fazMatricula", que se encarregará de estabelecer a verdadeira relação entre os objetos das classes relacionadas (Figura 19). Por fim, apresentamos na Figura 20 o diagrama de classes completo, resultante da análise orientada a objetos do sistema de controle de orientações e defesas de TCC.



ra 20 *Diagrama de Classes do sistema de orientações e defesas de TCC (UML).*