Modelagem do Sistema cadastro de clientes em UML.

SENAI - UC12 – SA2

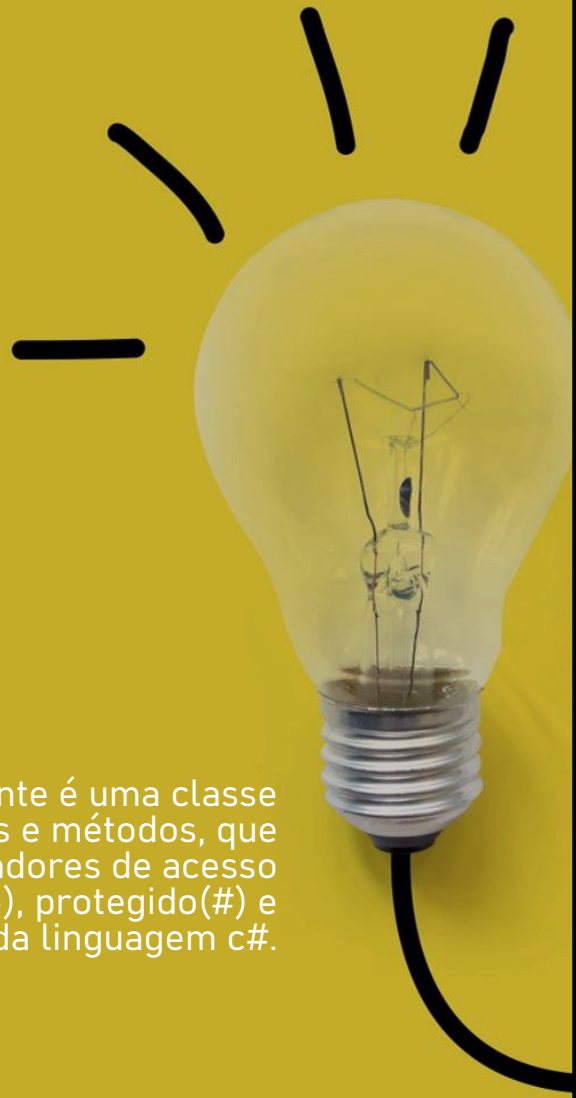
Aluno: Marcelo Carneiro Marques

Consiste na identificação de entidades envolvidas e o seu relacionamento dentro do contexto da aplicação.

Através de um diagrama de classes , foi possível relacionar as entidades envolvidas por meio de uma estrutura gráfica, demonstrando como cada um de seus componentes estão interligados.

Estrutura dos componentes.

Foi utilizado a ferramenta gratuita chamada draw.io, para a construção dos componentes, com base num sistema orientado a objetos.



Cada componente é uma classe constituída de atributos e métodos, que podem ter modificadores de acesso público(+), privado(-), protegido(#) e interno(~) , no caso da linguagem c#.

Estrutura dos componentes.

Os atributos são utilizados para armazenar informações, enquanto que os métodos são usados para realizar determinada ação. O ideal é que os atributos sejam privados e os métodos públicos, permitindo o encapsulamento dos dados, impedindo desta forma o acesso e alteração do conteúdo dos atributos de forma não autorizada.

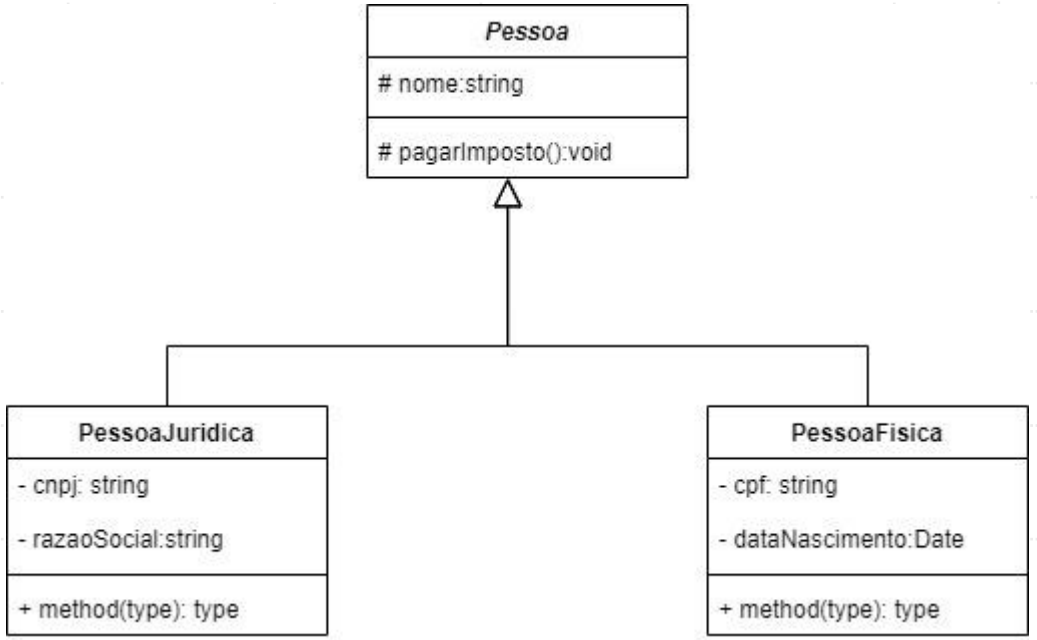
Através do encapsulamento, poderá ser realizada determinada ação, sem que sejam revelados os detalhes da implementação.

Modelagem de clientes

Inicialmente, foram criadas 2(duas) classes denominadas: Pessoa Física e Pessoa Jurídica com os seus atributos especificados. Contudo, pôde-se perceber que o atributo **nome** apareceu de forma repetida em ambas as classes. E isso é algo indesejável durante a modelagem. Para resolver esta questão, foi criada uma **classe abstrata (genérica) chamada *Pessoa (componente pai)*,** em que foi criado o atributo **nome**. Desta forma, as classes Pessoa Física e Jurídica passaram a ser componentes filhos, herdando o atributo nome do componente pai. Consequentemente, não foi mais necessário usar o atributo nome nas classes filhas. Similarmente, na atividade é solicitado que ambos (Pessoa Física e Jurídica) devam possuir a opção para ***pagar imposto***. Desta forma, foi criado um método na classe ***Pessoa*** (classe pai) para que ambas as classes filhas pudessem também herdar esse método. Nesta situação, foi estabelecida uma condição de **herança.**

Estrutura da modelagem

Observe que a classe *Pessoa* por ser abstrata é representada em *itálico* na linguagem UML. Esta é denominada uma classe pai em relação à Pessoa Jurídica e Pessoa Física (classes filhas) . Essa relação de herança, é representada através do uso de segmentos que ligam as classes filhas por meio de uma seta até a classe pai. Perceba que na classe Pessoa , antes de cada atributo e método, existe um modificador de acesso (# ou protected), que representa que os mesmos serão acessados somente pelas classes filhas.



Estrutura da modelagem

Na atividade foi informado que tanto Pessoa Física como Jurídica devam possuir um endereço, indicando se é comercial ou residencial.

Inicialmente foi pensado em por um único atributo chamado endereço do tipo string dentro da classe Pessoa. Contudo, é notório que endereço também possui atributos próprios que fazem parte de sua semântica tais como: logradouro, número, bairro, entre outros. Por isso, endereço é um atributo composto. Seria complicado gerir tanta informação, caso fosse usado um único atributo, até mesmo para construções futuras em um banco de dados, onde ficaria muito custoso localizar determinadas informações, ficando o código desorganizado.

Pensando nisso, foi então criada uma classe chamada **Endereço**, que passou a ter os atributos que a compõem. Mas e quanto ao relacionamento? Aonde ela poderá ser “encaixada”?

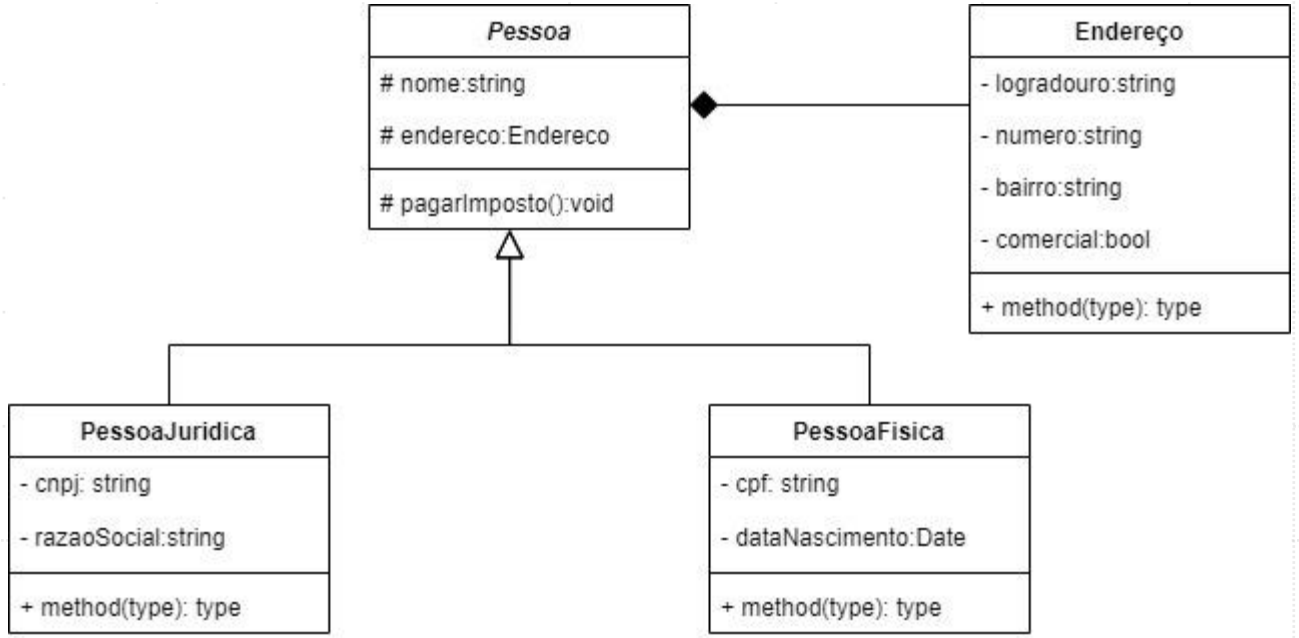
Como a classe Pessoa é uma classe genérica e abstrata, sendo , portanto, uma classe pai, qualquer atributo que seja posto nesta classe, será herdado automaticamente pelas classes filhas, evitando a repetição. Desta forma, foi relacionado a classe Endereço com a classe Pessoa. O atributo endereço da classe Pessoa passou a ser *do tipo Endereço*.

continua...

Estrutura da modelagem

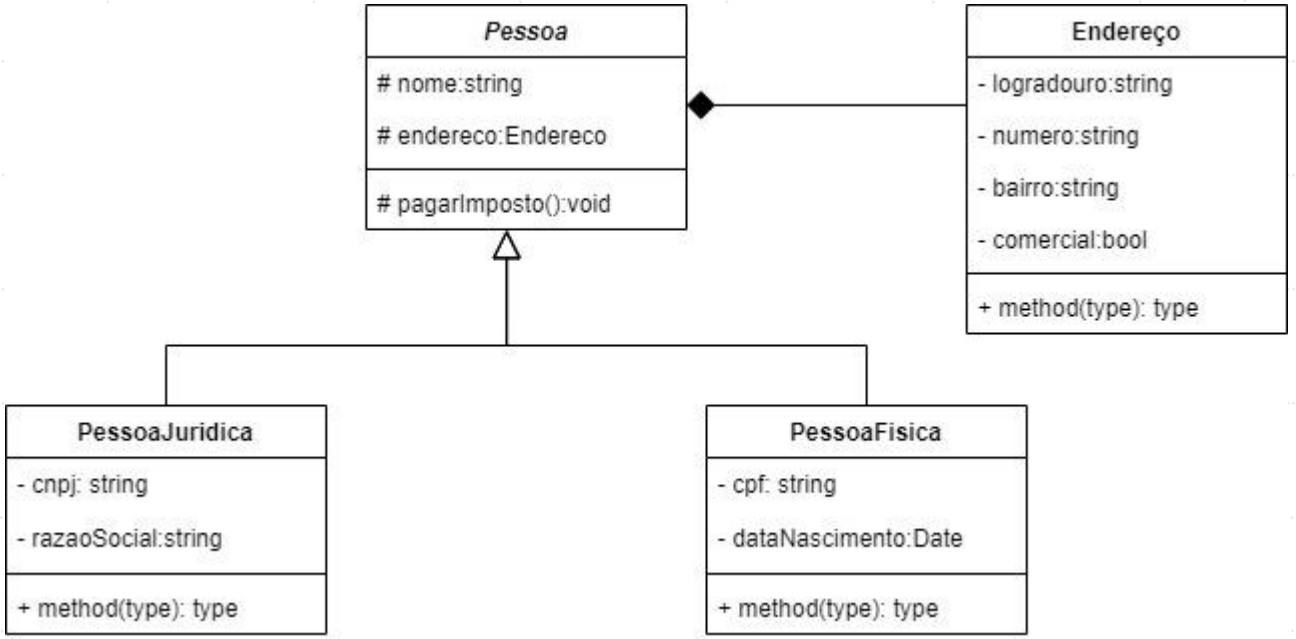
Toda pessoa deve possuir um endereço. Então um endereço deve fazer parte da classe Pessoa. Logo, trata-se de uma relação TODO-PARTES, onde o todo não faz sentido sem as partes. E da mesma forma, a parte também não faz sentido sem o todo. Neste caso, pessoa não tem razão sem endereço, e um endereço, deve possuir um ator a ele atrelado (cliente, pessoa, etc), não podendo existir sozinho.

Para representar a composição, foi criado um segmento da classe Endereço(parte) em direção à classe Pessoa (todo), tendo um losango preto em sua extremidade.



Estrutura da modelagem

E para finalizar, deve-se indicar se o endereço é comercial ou residencial. Com esse intuito , foi adicionado uma atributo booleano chamado *comercial*, que poderá ter apenas dois valores: verdadeiro ou falso. Quando o atributo comercial for falso, subtende-se que o endereço é residencial. Também poderia ter sido usado um atributo residencial em seu lugar, seguindo a mesma lógica de raciocínio. Outros tipos também não devem ser descartados como , por exemplo, a enumeração, mas deve-se ressaltar que o booleano é mais simples de armazenar já que são usados apenas os valores verdadeiro ou falso. Observe que na classe Endereço poderiam ser completados mais atributos para a classe. Este exercício, apenas consistiu numa representação do modelo, e apenas foi seguido o que havia sido solicitado. Obviamente, que posteriormente, terão de ser criados métodos públicos de acesso para obter/definir informação dos atributos do objeto que sejam privados.



Definições conceituais nos relacionamentos

Associação: Ocorre quando existe um relacionamento entre objetos independentes entre si em que um não dependa do outro para existir. Pode ser considerada uma relação de *muitos pra muitos.*
Ex: Relação entre um professor e alunos. Um aluno pode ter vários professores e um professor pode ter vários alunos. Professores podem existir sem alunos e alunos podem existir sem professores.

Agregação: Também caracteriza uma associação, em que um objeto (todo) agrega "partes" e o todo NÃO determina a existência de suas partes, sendo consideradas *classes independentes*.
Ex: Turma e Aluno.
Se uma turma deixar de existir, os alunos irão continuar existindo.

Definições conceituais nos relacionamentos

Composição: É um tipo especial de agregação, em que o objeto (todo) compõe as partes e o todo determina a existência de suas partes, sendo consideradas classes dependentes.

Ex: Pessoa e Endereço
Uma pessoa deve possuir um endereço. Se uma pessoa(todo) deixar de existir, automaticamente o endereço (parte) também não faz mais sentido existir.

Aluno: Marcelo Carneiro Marques
Curso SENAI.
UC12 – SA2
14/01/2022.
Profº Odirlei Sabella de Assis.
Profº Thiago Rocha do Nascimento.