UML – Criando Diagramas Eficientes

Prof: Nilson Júnior

Agenda

- O que é UML
- História
- Diagramas UML
- Diagrama de Caso de Uso
- Diagrama de Classes
- Ferramentas de Modelagem

O que é e por que usar UML?

- UML – Unified Modeling Language
- Definição

"É uma família de **notações gráficas**, apoiada por um metamodelo único, que ajuda na **descrição** e no **projeto** de sistemas de software, particularmente daqueles construídos utilizando o estilo orientado a objetos."

Martin Fowler

UML - Diagramas

- Lista de Diagramas

Diagrama	Objetivo	Grupo Diagrama
Classes	**Classe, características e relacionamentos.**	**Estrutural**
Componentes	Estrutura e conexão de componentes.	Estrutural
Estruturas Compostas	Decomposição de uma classe em tempo de execução.	Estrutural
Instalação	Distribuição de artefatos nos nós.	Estrutural
Objetos	Exemplo de configurações de instâncias.	Estrutural
Pacotes	Estrutura hierárquica em tempo de compilação.	Estrutural
Casos de Uso	**Como os usuários interagem com um sistema.**	**Comportamental**
Atividades	Comportamento procedimental e paralelo.	Comportamental
Máquinas de Estado	Como os eventos alteram um objeto no decorrer de sua vida.	Comportamental
Sequência	Interação entre objetos; ênfase na sequência.	Interação
Comunicação	Interação entre objetos; ênfase nas ligações.	Interação
Interatividade	Mistura de diagrama de sequência e de atividades.	Interação
Tempo	Interação entre objetos; ênfase no sincronismo.	Interação

UML – Diagrama de Caso de Uso

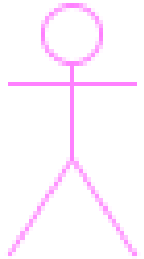
"Documento narrativo que descreve a sequência de eventos de um ator que usa um sistema para completar um processo."

Ivar Jacobson

- Representa a interação entre um usuário (humano ou sistema) e o sistema.
- Não descreve como o software deverá ser construído, mas sim como ele deverá se comportar quando estiver pronto.
- Corresponde a um conjunto de ações com um objetivo comum.

Ator

- Humano ou entidade.
- Interage com o sistema.
- Iniciam o sistema.
- Fornecem dados.
- Usam as informações do sistema.



Caso de Uso

- Unidade de um trabalho significante.
- Representa um processo.
- Iniciado por um ator ou outro caso de uso.
- Exemplos: “Login para o sistema”, “Registrar no sistema”, “Criar pedidos”, etc.

<<include>> e <<extend>>

- <<include>>
  - Relacionamento com outro caso de uso que sempre será executado.

- <<extend>>
  - Relacionamento com outro caso de uso que pode ou não ser executado.

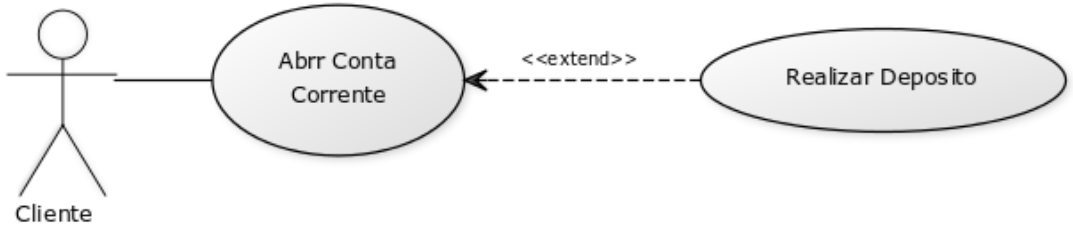


Diagrama de Caso de Uso

Descrição

Pagamento de Serviço

Cenário Principal de Sucesso:
1. O usuário acessa o sistema
2. O usuário pesquisa o serviço a ser pago
3. O sistema apresenta as informações do serviço
4. O usuário inicia o processo de pagamento
5. O sistema envia a confirmação do pagamento ao prestador do serviço
6. O sistema encerra o processo de pagamento

Extensões:
1a. Usuário não autorizado
  1a.1 O usuário não possui perfil para realizar pagamentos
  1a.2 O usuário é direcionado ao passo 6.
3a. Serviço não finalizado
  3a.1 O sistema apresenta que o serviço não foi finalizado
  3a.2 O usuário é direcionado ao passo 6.

Diagrama

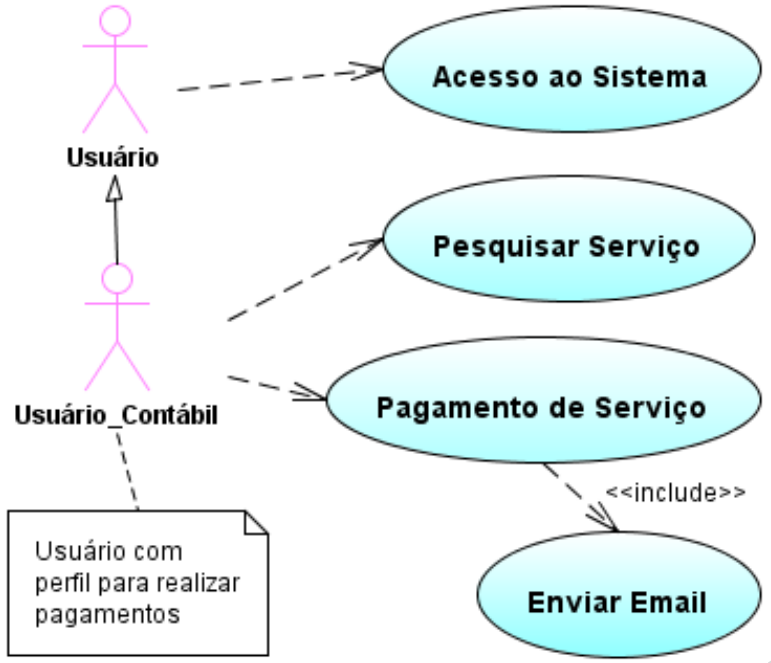
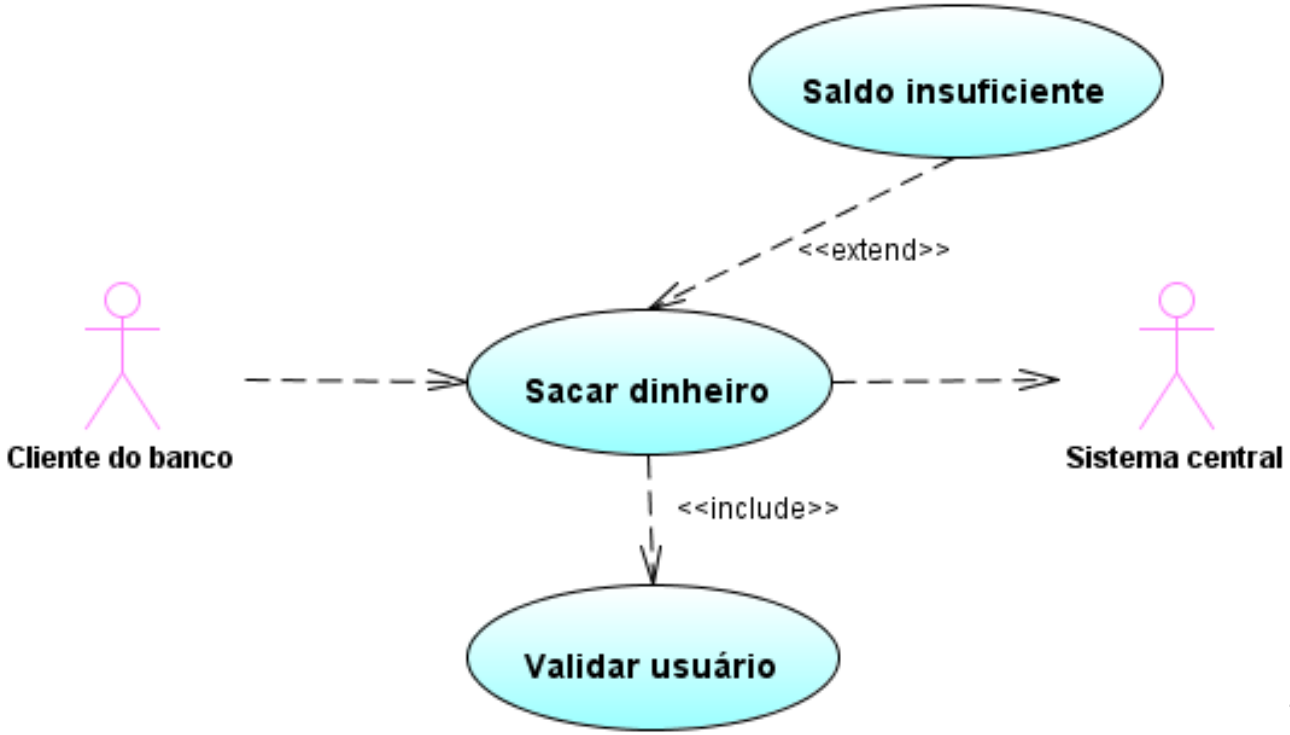


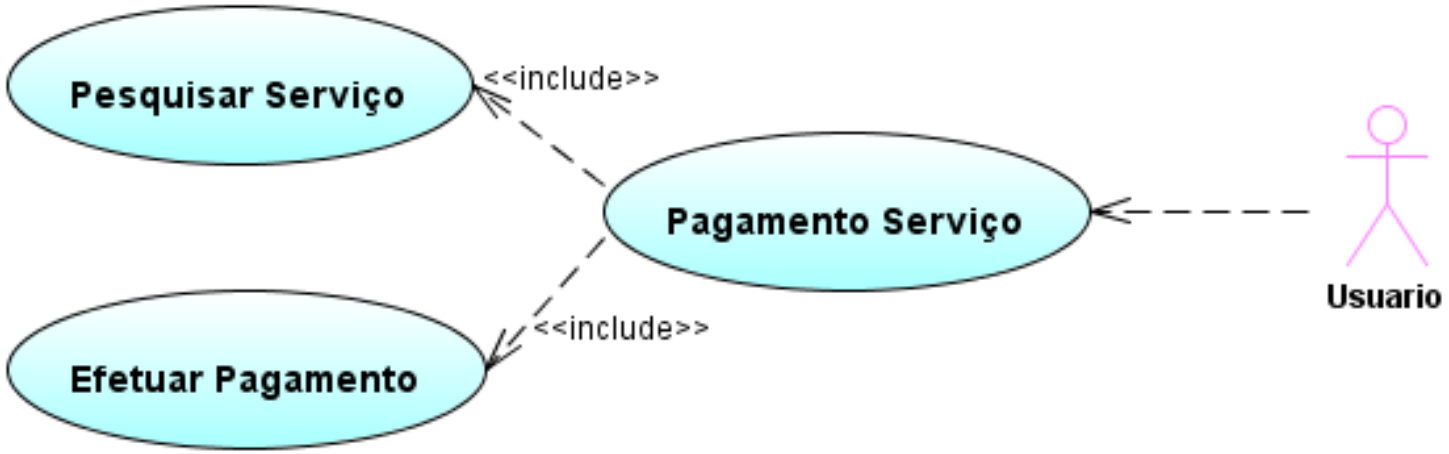
Diagrama de Caso de Uso

- Exemplo de Caso de Uso para sacar dinheiro



Resposta do exemplo prático

- Sistema de Pagamento de Serviços, realizar pesquisa de serviços



O que colocar no diagrama de Caso de Uso

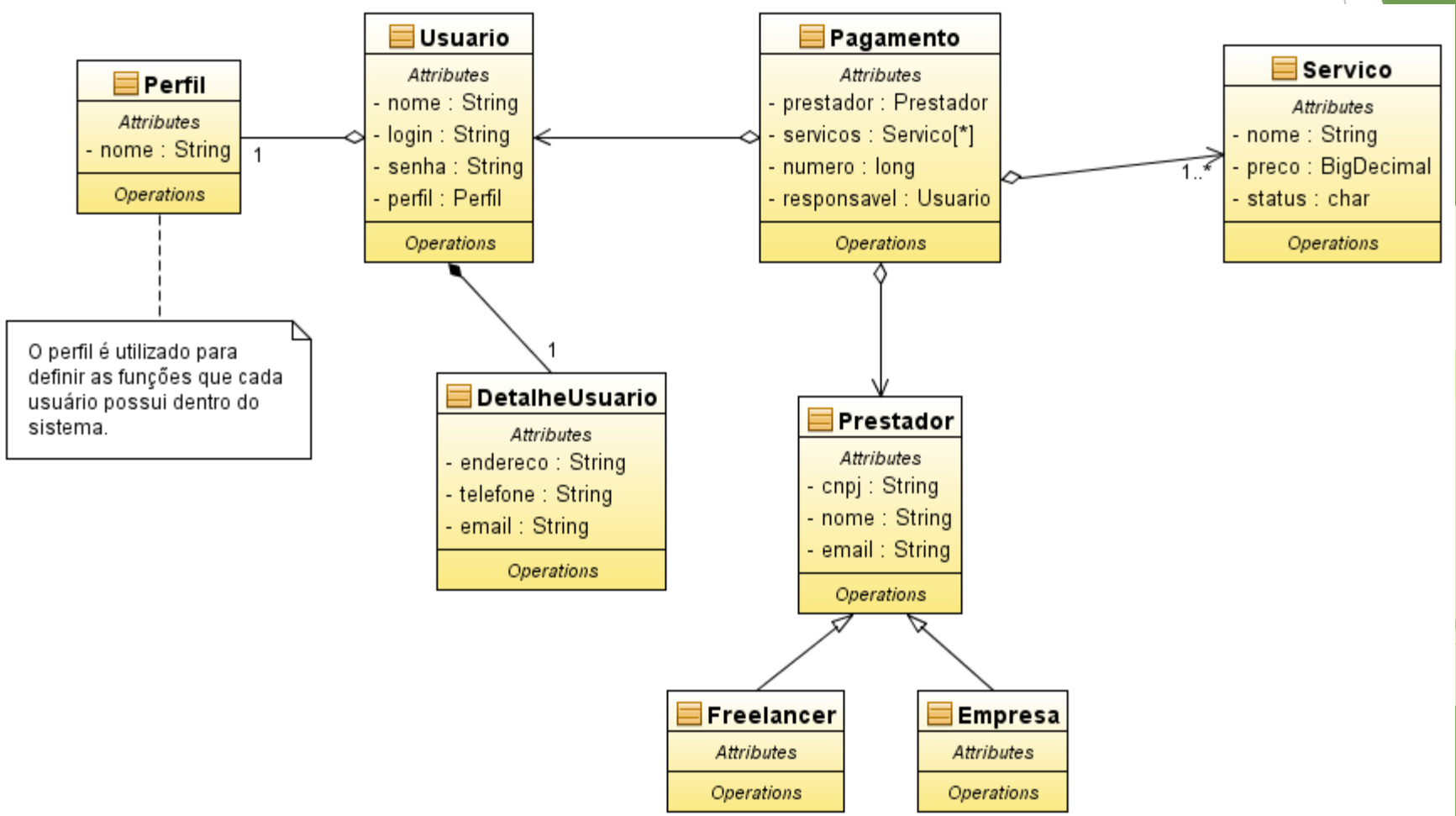
- Melhor fazer menos do que fazer demais.
- Breve e fácil de ler.
- Preferência na descrição textual.
- Limitar os relacionamentos com **<<include>>** e **<<extend>>**.

O que não colocar no diagrama de Caso de Uso

- Textos longos.
- Muitas extensões.
- Todos diagramas se chamando.
- Todas as ações CRUD separadas.
- Detalhes da tela (botões, combos, links, etc).
- Não é um fluxograma!

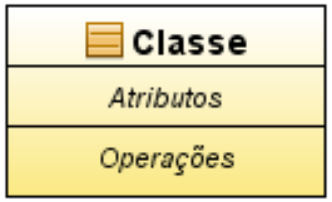
Atenção: Não relacione o caso de uso com as classes do sistema.

UML – Diagrama de Classes



Estrutura da classe

- Uma classe em UML possui três partes:
  - Nome da Classe
  - Atributos
  - Operações

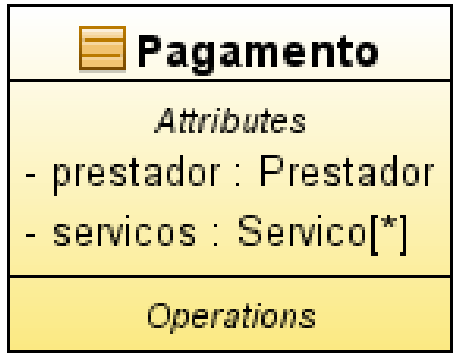


- Podemos abreviar a declaração da classe, caso não influencie o entendimento do diagrama:

Atributos

- Um atributo é formado por:

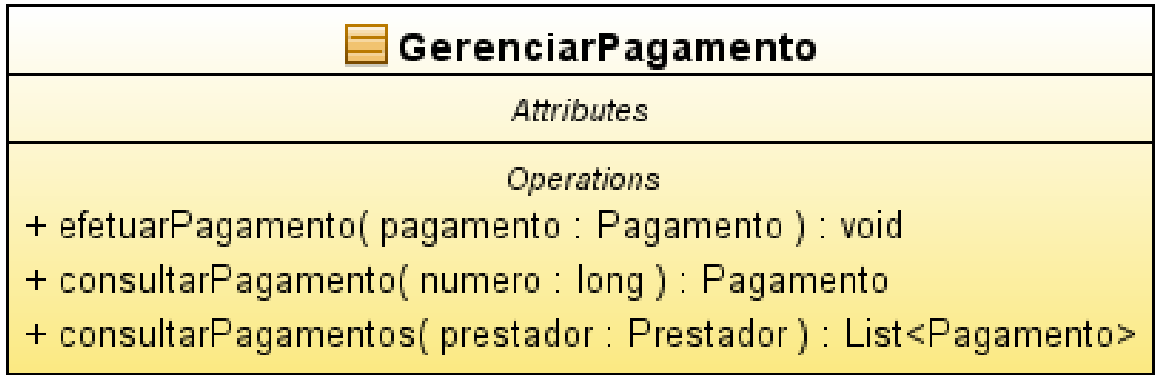
visibilidade nome **:** tipo [multiplicidade] = valor inicial {propriedades}



Operações

- Uma operação é formada por:

  visibilidade nome (parâmetros) : tipo de retorno {propriedades}



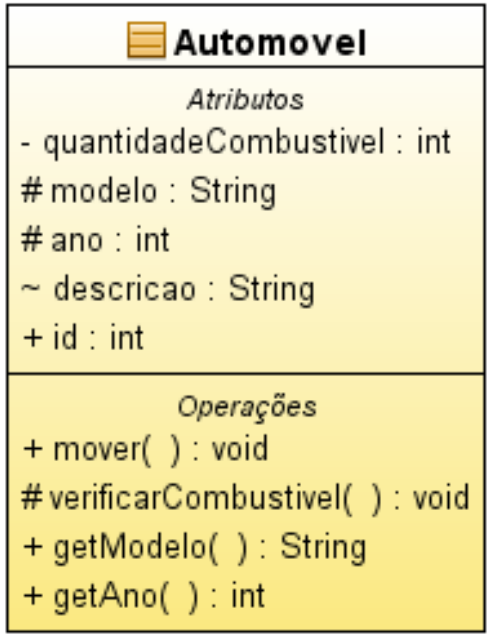
- O parâmetro de um método é formado por:

  nome : tipo [multiplicidade] = valor inicial

Visibilidade

- Podemos definir as seguintes visibilidades em atributos e operações:

  - `-` private
  - `~` default
  - `#` protected
  - `+` public

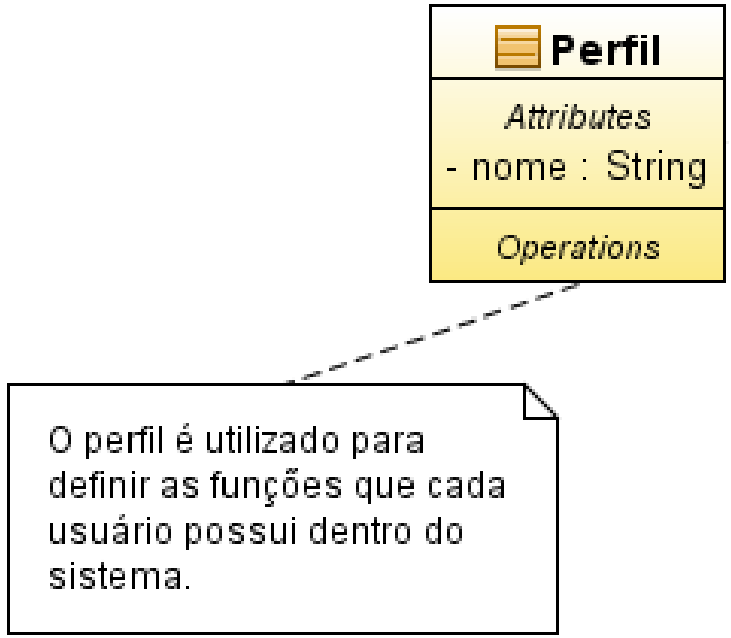


Atributos e operações estático

- Podemos definir atributos e operações como sendo estáticos, ou seja, são referentes a classe e não aos seus objetos.

Comentário

- Os comentários ou notas são utilizados para adicionar mais informações ao diagra

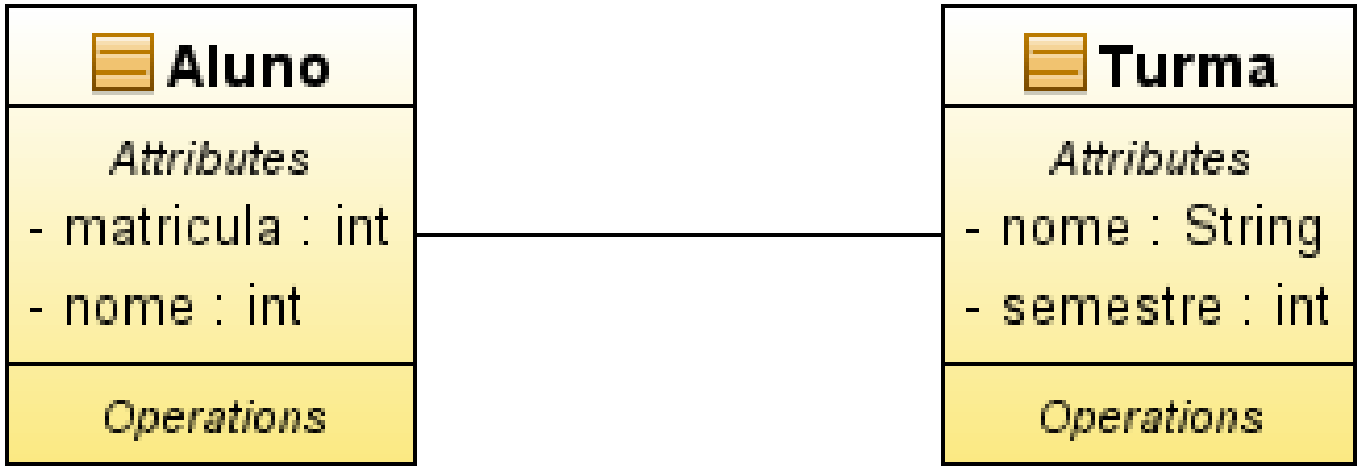


Associações

- Utilizado para representar o relacionamento entre classes, as associações podem ser:
  - Associação
  - Agregação
  - Composição
- As classes que fazem parte de um relacionamento também são chamadas de TODO (responsável pelo relacionamento) e PARTE (usado pelo relacionamento).

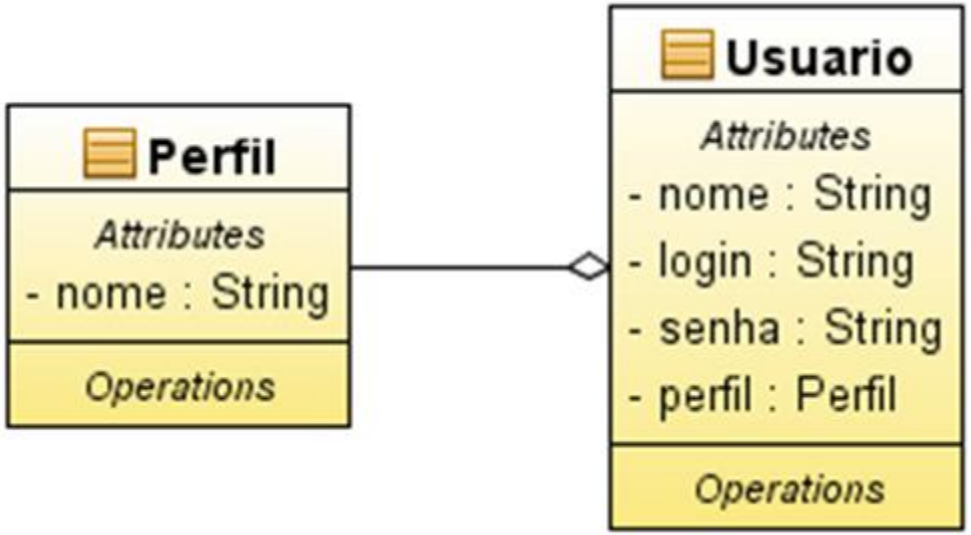
Associação

- Relacionamento simples entre duas classes:



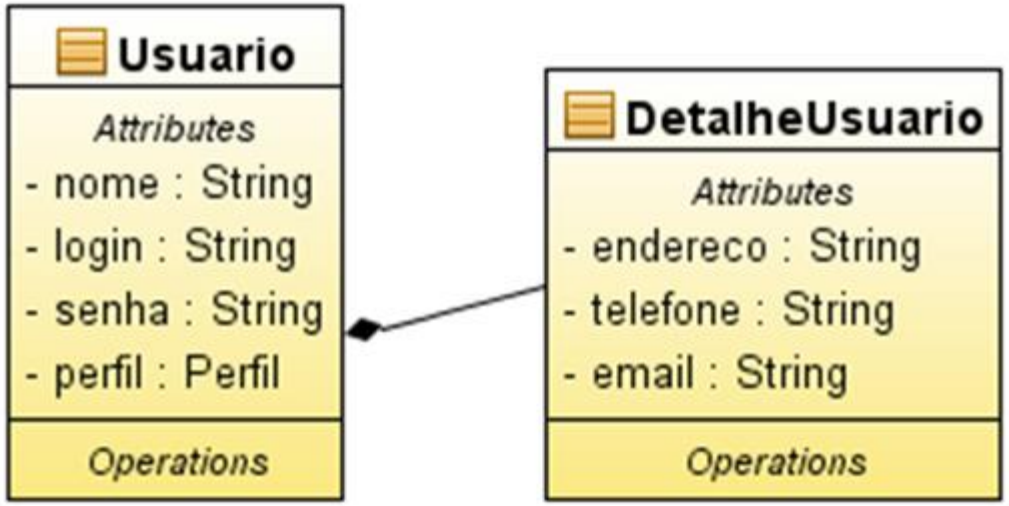
Agregação

- Informa que uma classe faz parte de outra classe, mas não de forma exclusiva.



Composição

- Informa que uma classe faz parte de outra classe de forma exclusiva.



Agregação x Composição

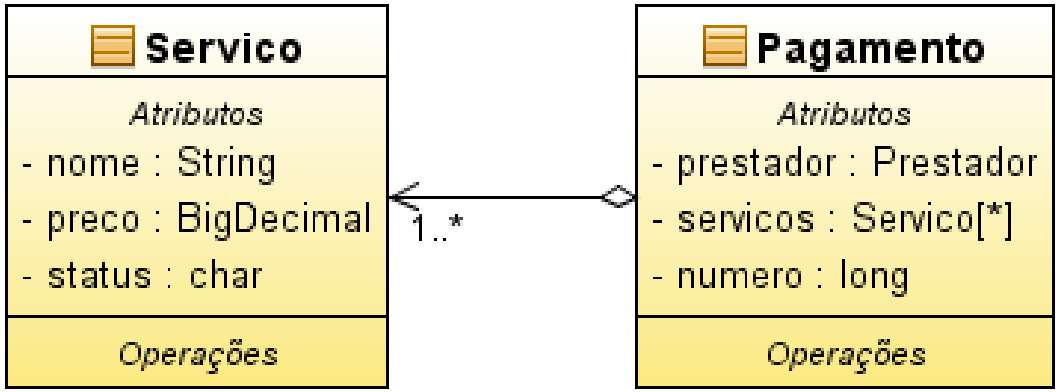
- A diferença entre ambos é:

Agregação – se excluir a classe responsável pelo relacionamento, não deve excluir a classe que ele possui relacionamento.

Composição – se excluir a classe responsável pelo relacionamento, então deve excluir a classe que ele possui relacionamento.

Multiplicidade

- A multiplicidade é utilizada para definir a quantidade de objetos devem ser criados:
  - **0 .. 1** (zero ou um)
  - **1** (um)
  - ***** (zero ou muitos)



Exemplo

- Crie o diagrama de classes UML para a seguinte figura: