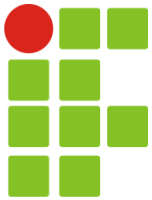


INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO NORTE
CAMPUS JOÃO CÂMARA

UML – UNIFIED MODELING LANGUAGE

Nickerson Fonseca Ferreira
nickerson.ferreira@ifrn.edu.br

O que é??

- A UML (*Unified Modeling Language*) é uma linguagem gráfica utilizada para:
  - Visualizar
  - Especificar
  - Construir
  - Documentar
  - Comunicar

    Os artefatos de sistemas complexos
- A Linguagem é composta por: vocabulário + regras de combinação
- Atualmente na versão 2.5

Por quê utilizar??

- Melhor compreensão do sistema que está sendo desenvolvido.
- Visualizar o sistema.
- Documentar tomadas de decisão.
- Especificar comportamento ou a estrutura de um sistema.
- Melhorar na comunicação com o cliente.

UML não é...

- Um processo
- Uma metodologia
- Análise e projeto OO
- Regra de projeto

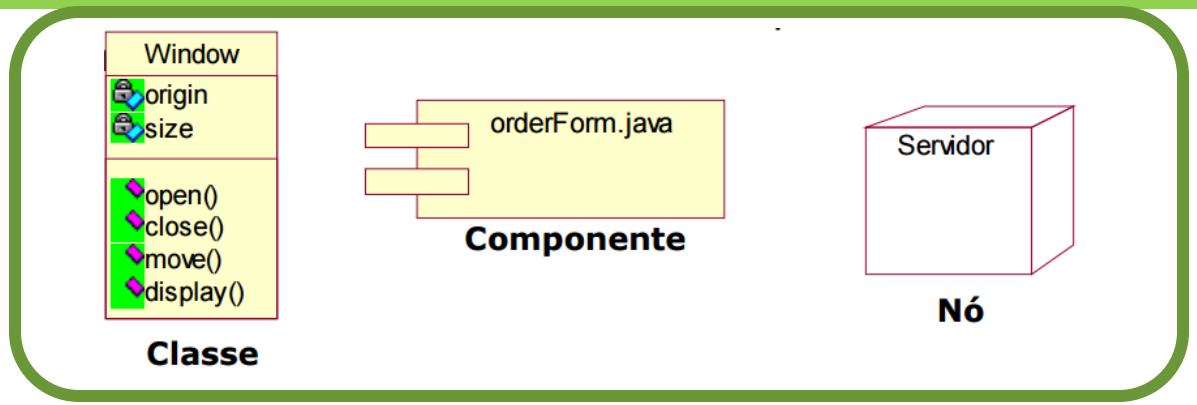
Blocos de construção

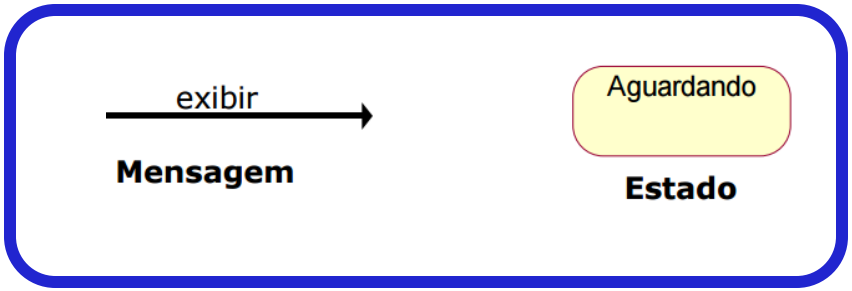
- É o principal elemento da UML.
- Possui 3 tipos:
  - Itens: são abstrações
  - Relacionamentos: realizam as ligações entre os itens
  - Diagramas: agrupam coleções de itens

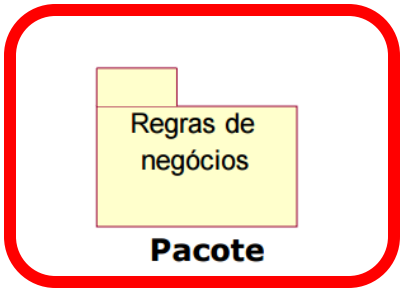
Itens da UML

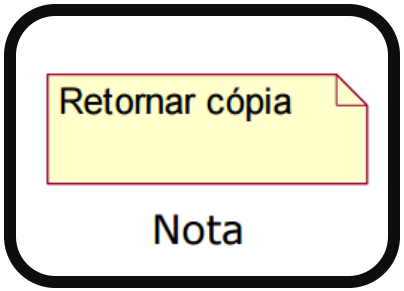
- Estruturais: parte estática, representa os elementos conceituais ou físicos.
  - Ex: Classes, interfaces, caso de uso, componentes e nós.
- Comportamentais: parte dinâmica dos modelos, representando comportamentos no tempo e no espaço.
  - Ex: interação e estado.
- De agrupamento: parte organizacional. Blocos em que os modelos podem ser decompostos - pacotes.
- Anotacionais: parte explicativa dos modelos – notas explicativas.

Itens da UML


Window
origin
size
open()
close()
move()
display()
Classe
orderForm.java
Componente
Servidor
Nó


exibir
Mensagem
Aguardando
Estado


Regras de negócios
Pacote

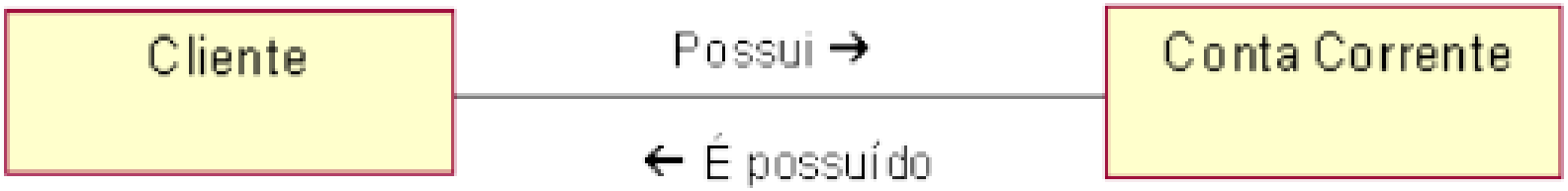

Retornar cópia
Nota

Relacionamentos

- Associação e Agregação
- Generalização/Especialização
- Dependência

Associação

- É um relacionamento estrutural que descreve um conjunto de ligações, em que as ligações são conexões entre objetos.
- Representada por uma linha entre os itens.

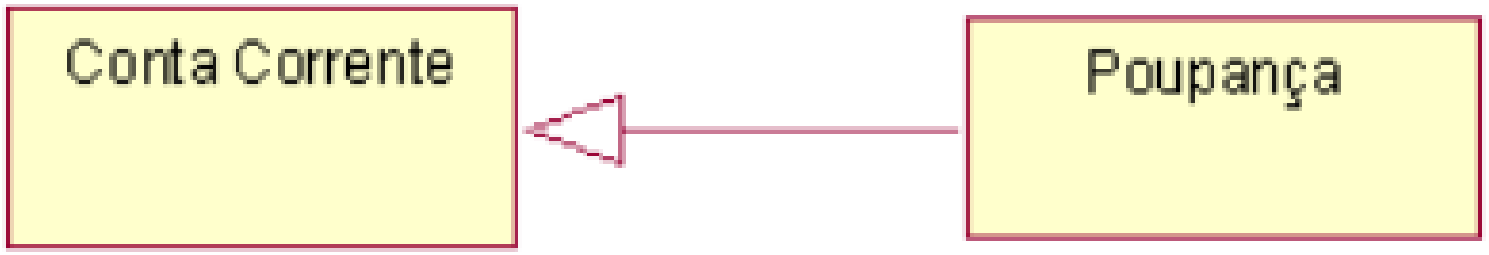


Agregação

- É um tipo particular de associação.
- A agregação indica que uma das classes do relacionamento é uma parte, ou está contida em outra classe.

Generalização/Especialização

- A generalização é um relacionamento entre um elemento geral e um outro mais específico.
- O elemento mais específico possui todas as características do elemento geral e contém ainda mais particularidades.
- Um objeto mais específico pode ser usado como uma instância do elemento mais geral.



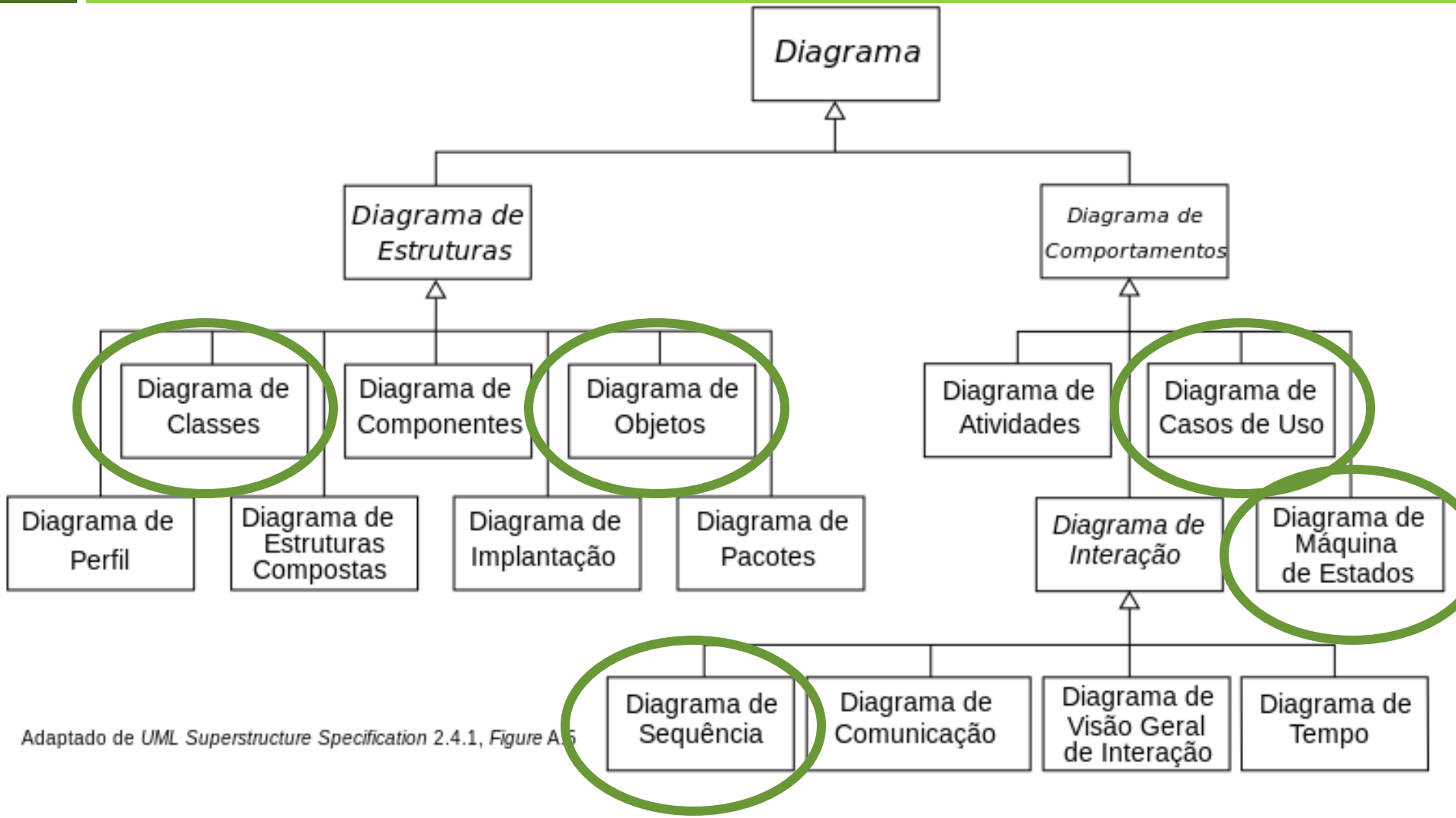
Dependência

- O relacionamento de dependência é uma conexão semântica entre dois modelos de elementos, um independente e outro dependente.
- Uma mudança no elemento independente irá afetar o modelo dependente.

Diagramas

- Apresentações gráficas de um conjunto de elementos, geralmente representadas como gráficos de vértices (itens) e arcos (relacionamentos).
- A UML possui, atualmente, 14 diagramas.
- Esses diagramas estão subdivididos em duas grandes categorias:
  - Estruturais
  - Comportamentais

Diagramas



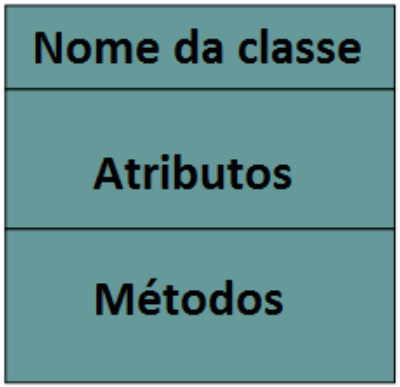
Adaptado de *UML Superstructure Specification* 2.4.1, *Figure* A.5

Diagrama de Classes

- Diagramas de classe são a espinha dorsal da maioria dos métodos orientados a objeto, inclusive UML.
- Descrevem a estrutura estática do sistema (entidades e relacionamentos).
- Elementos principais:
  - Classes
  - Relacionamentos

Diagrama de Classes

- Uma **CLASSE** é a descrição de um conjunto de objetos que compartilham os mesmos atributos, métodos, relacionamentos...
- Os **ATRIBUTOS** são propriedades de uma classe e assumem valores para cada instância.
- Os **MÉTODOS** são os serviços que podem ser requisitados para realizar determinada tarefa.



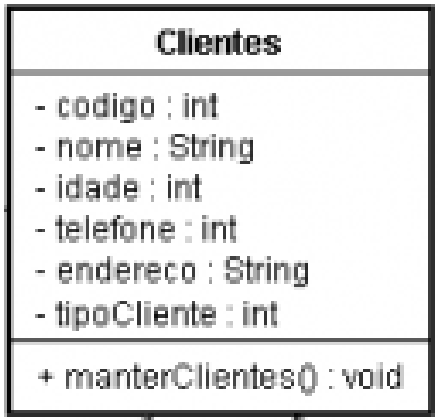


Diagrama de Classes

- **Atributos**
  - Nome_atrib : tipo [= valor_inicial]
    - Idade: int = 0
    - DtNascimento : Data
- **Operações**
  - Nome_oper (parâmetros) : tipo_retorno
    - setIdade(id:int) : void
    - getIdade() : int

Diagrama de Classes

- **Visibilidade**
  - Privado (**-**)
    - - idade : int
    - - calcularDias(idade: int)
  - Público (**+**)
    - + idade : int
    - + calcularDias(idade: int)
  - Protegido (**#**)
    - # idade : int
    - # calcularDias(idade: int)

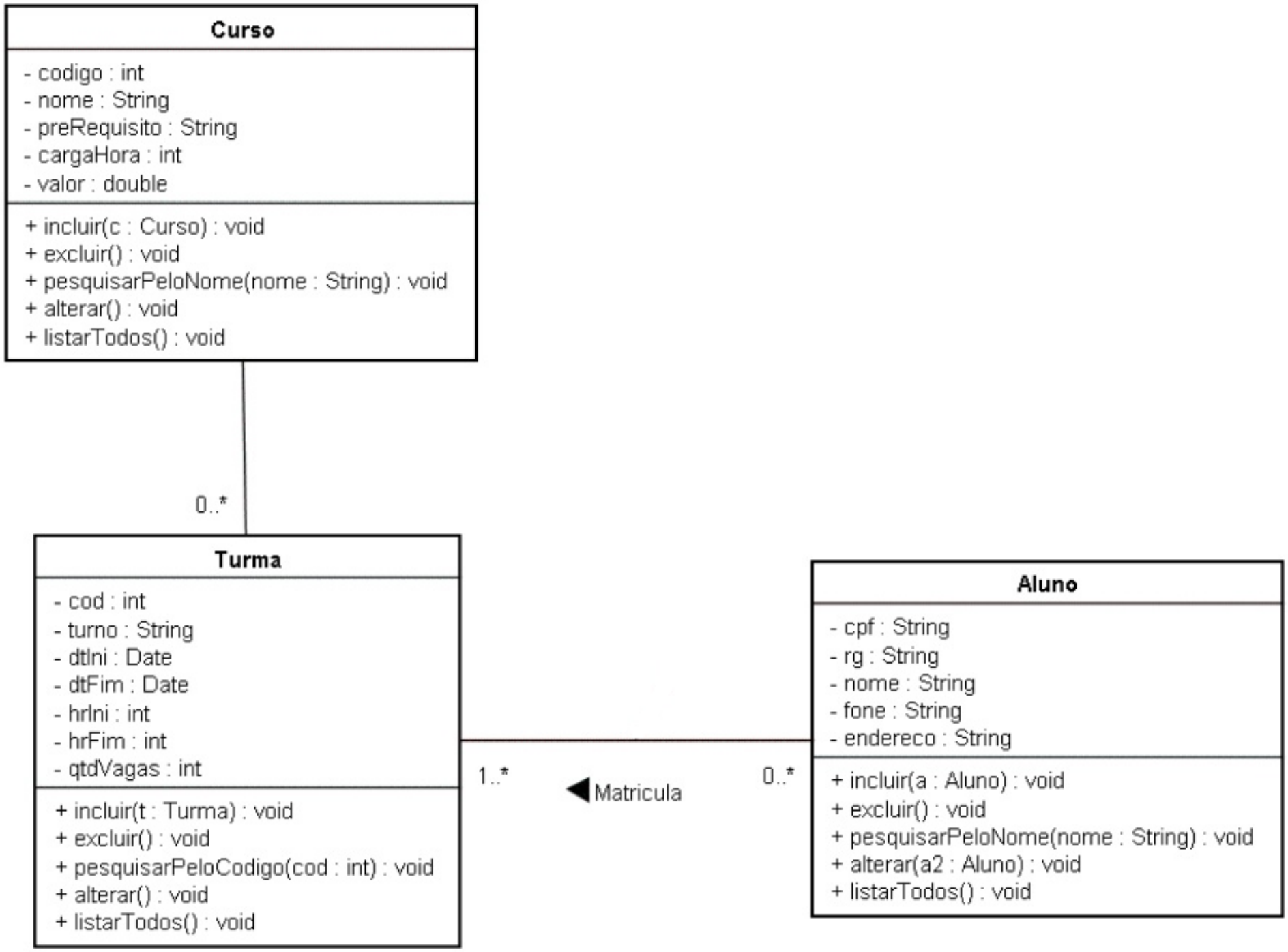

Curso
- codigo : int
- nome : String
- preRequisito : String
- cargaHora : int
- valor : double
+ incluir(c : Curso) : void
+ excluir() : void
+ pesquisarPeloNome(nome : String) : void
+ alterar() : void
+ listarTodos() : void
0..*
Turma
- cod : int
- turno : String
- dtIni : Date
- dtFim : Date
- hrIni : int
- hrFim : int
- qtdVagas : int
+ incluir(t : Turma) : void
+ excluir() : void
+ pesquisarPeloCodigo(cod : int) : void
+ alterar() : void
+ listarTodos() : void
1..*
Matricula
0..*
Aluno
- cpf : String
- rg : String
- nome : String
- fone : String
- endereco : String
+ incluir(a : Aluno) : void
+ excluir() : void
+ pesquisarPeloNome(nome : String) : void
+ alterar(a2 : Aluno) : void
+ listarTodos() : void

Diagrama de Objetos

- Descrevem a estrutura estática de um sistema **em um determinado momento.**
- Podem ser usados para testar a precisão dos diagramas de classe.

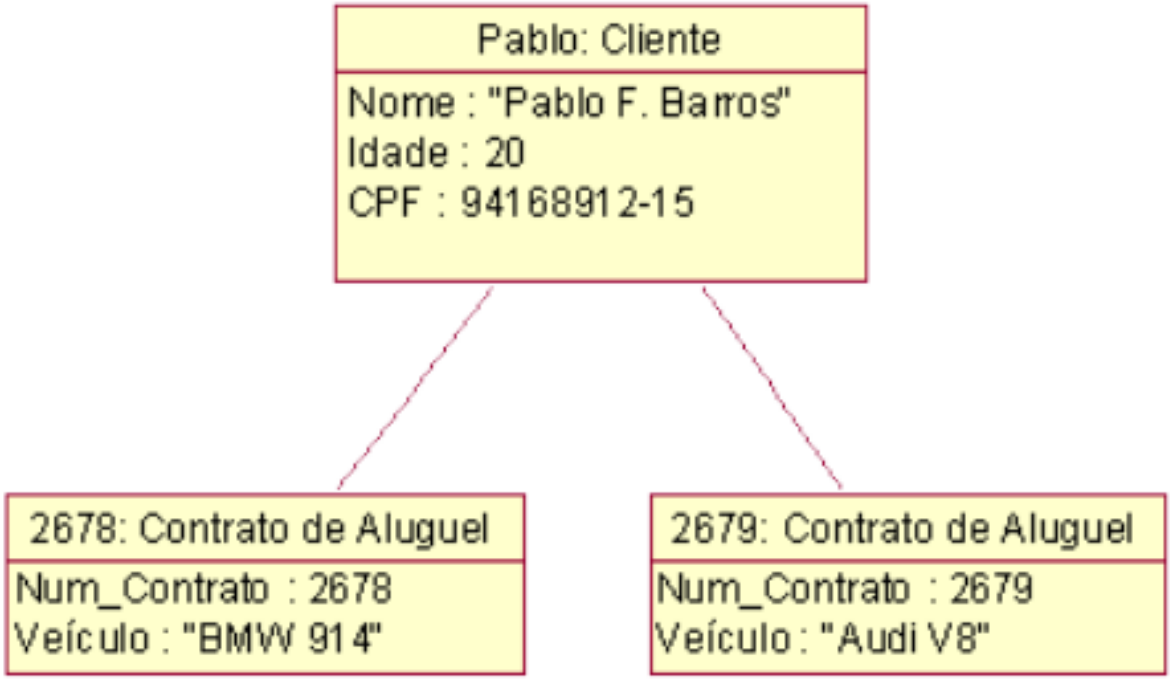


Diagrama de Caso de Uso

- Modelam a funcionalidade do sistema através de atores e casos de uso.
- Elementos principais:
  - Casos de Uso
  - Atores
  - Relacionamentos

Diagrama de Caso de Uso

- **CASOS DE USO** são serviços ou funções fornecidas pelo sistema aos seus usuários.
- São representados por um círculo com um título no centro.
- O título deve passar a ideia da tarefa que o caso de uso realiza.

Diagrama de Caso de Uso

- Os **ATORES** representam o papel de uma entidade externa ao sistema como um usuário, um hardware, ou outro sistema que interage com o sistema modelado.
- São representados por um boneco e possui um nome.

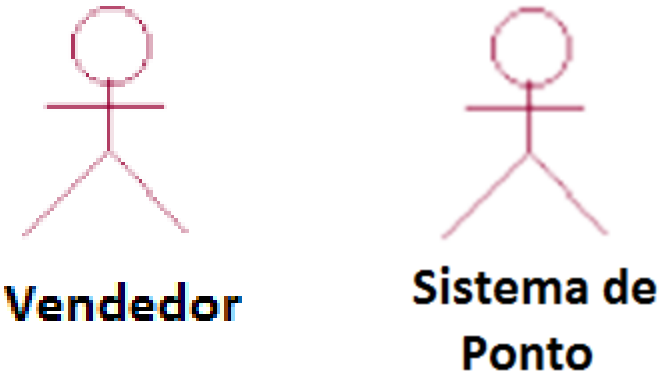


Diagrama de Caso de Uso

- Os relacionamentos entre os casos de uso possuem uma diferença, utilizam um estereótipo.
- Um estereótipo é uma extensão do vocabulário da linguagem UML. Ele permite a criação de novos tipos de blocos de criação específicos para resolução de problemas.
  - Ex: <<interface>>, <<includes>>, <<extends>>
- Os relacionamentos mais usados são:
  - Generalização
  - Inclusão (<<includes>>)
  - Extensão (<<extends>>)

Diagrama de Caso de Uso

- **Inclusão** (<<includes>>): é usado quando casos compartilham comportamento comum com outros UC.
  - O caso de uso é executado SEMPRE.
- **Extensão** (<<extends>>): é a utilização inversa da inclusão, e pode (não necessariamente) alterar o comportamento do UC que foi estendido.
  - O caso de uso pode ou não ser executado.

Diagrama de Caso de Uso

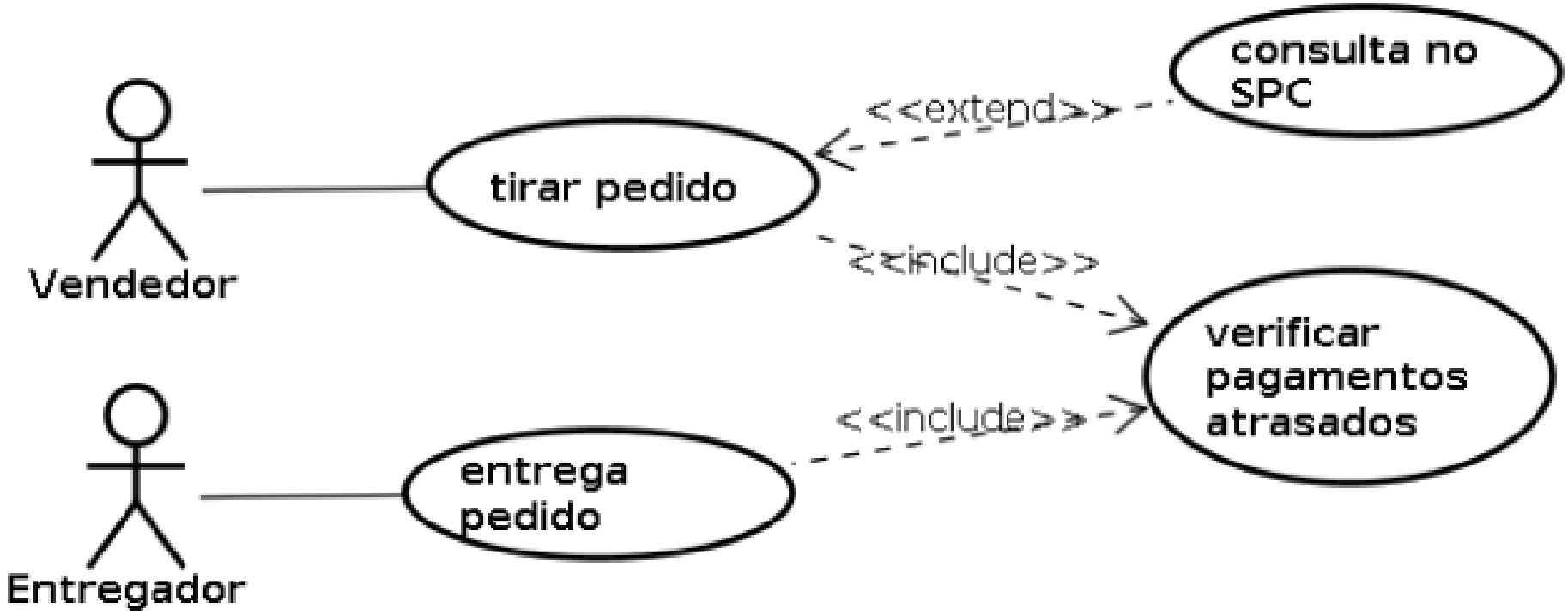


Diagrama de Estados

- O diagrama de estado é tipicamente um complemento para a descrição das classes.
- Este diagrama mostra todos os estados possíveis que objetos de uma certa classe podem se encontrar e mostra também quais são os eventos do sistemas que provocam tais mudanças.
- Diagramas de estado capturam o ciclo de vida dos objetos, subsistemas e sistemas.

Diagrama de Estados

- O diagrama de estado é tipicamente um complemento para a descrição das classes.
- Este diagrama mostra todos os estados possíveis que objetos de uma certa classe podem se encontrar e mostra também quais são os eventos do sistemas que provocam tais mudanças.
- Diagramas de estado capturam o ciclo de vida dos objetos, subsistemas e sistemas.
- Composto por:
  - Estados
  - Eventos
  - Sinais

Diagrama de Estados

- **Estados:** é uma condição que um objeto pode possuir durante seu ciclo de vida e enquanto satisfaça alguma condição.
- **Eventos:** é uma ocorrência significante que torna a transição habilitada
- **Sinais:** comunicação entre os objetos.

Diagrama de Estados

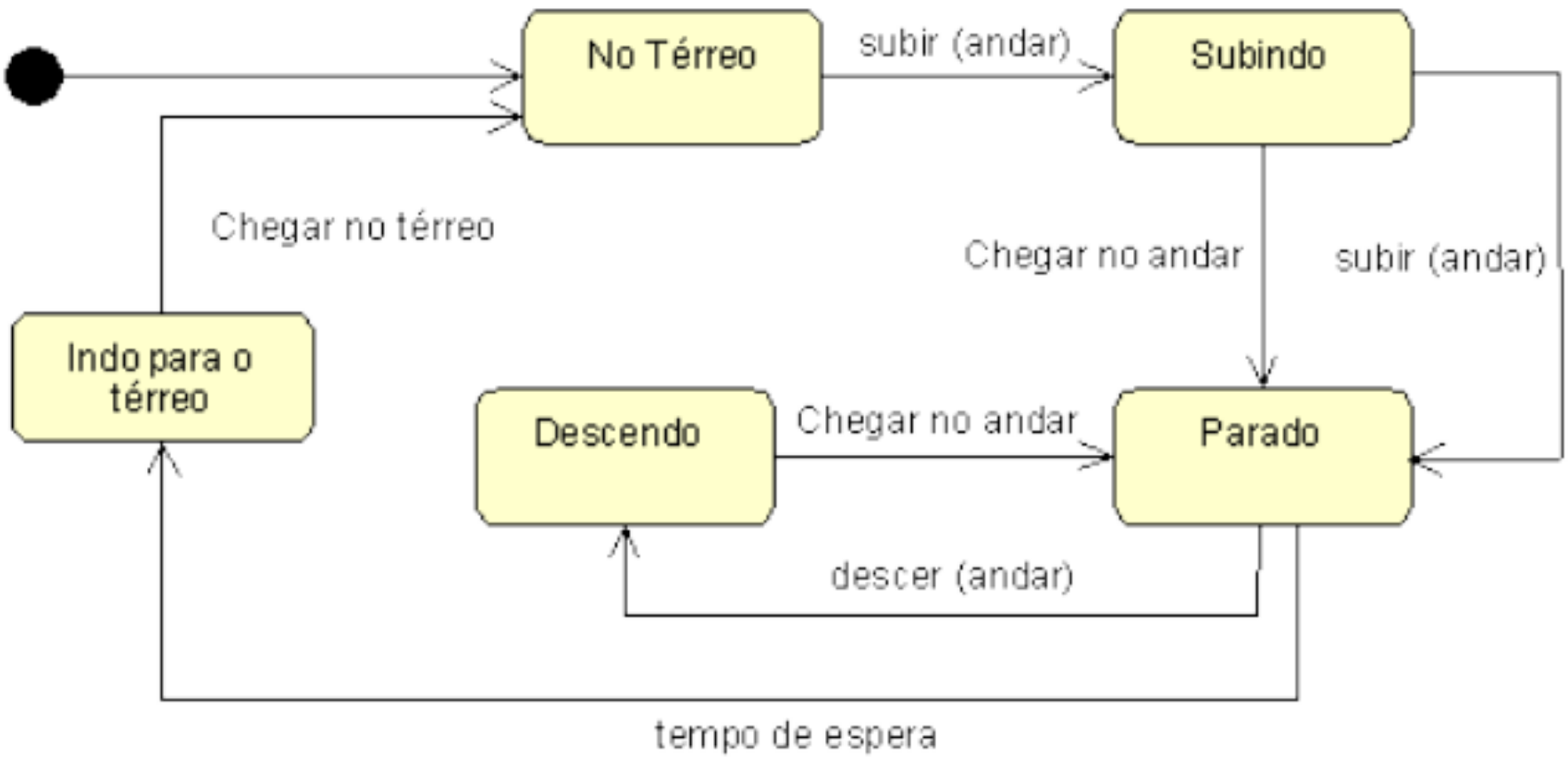


Diagrama de Sequência

- Descreve as interações entre as classes através das trocas de mensagens ao logo do tempo.
- Ele mostra a interação entre os objetos, alguma coisa que acontecerá em um ponto específico da execução do sistema.
- Diagramas de sequência possuem dois eixos: o eixo vertical, que mostra o tempo e o eixo horizontal, que mostra os objetos envolvidos na sequência de uma certa atividade.

Diagrama de Sequência