

Aula 6

Professores:

Cláudia Maria Lima Werner

Leonardo Gresta Paulino Murtas

Projeto OO na UML

Conteúdo:

- Descrição de casos de uso concretos
- Mapa de navegação
- Modelo de classes
- Modelo de seqüência
- Modelo de estados

Artefatos produzidos pelo POO

- Descrição de casos de uso concretos;
- Mapa de navegação;
- Modelo de classes (projeto);
- Modelos de seqüência/collaboração para métodos não triviais;
- Modelos de transição de estados para classes não triviais;

Descrição de casos de uso concretos

- A UML não define como se deve descrever um caso de uso;
- O modelo de casos de uso oferece somente uma visão geral do sistema;
- A descrição de casos de uso oferece uma visão detalhada de um caso de uso;
- Para POO, a descrição de casos de uso é um artefato importante como ponto de partida;

Descrição de casos de uso concretos

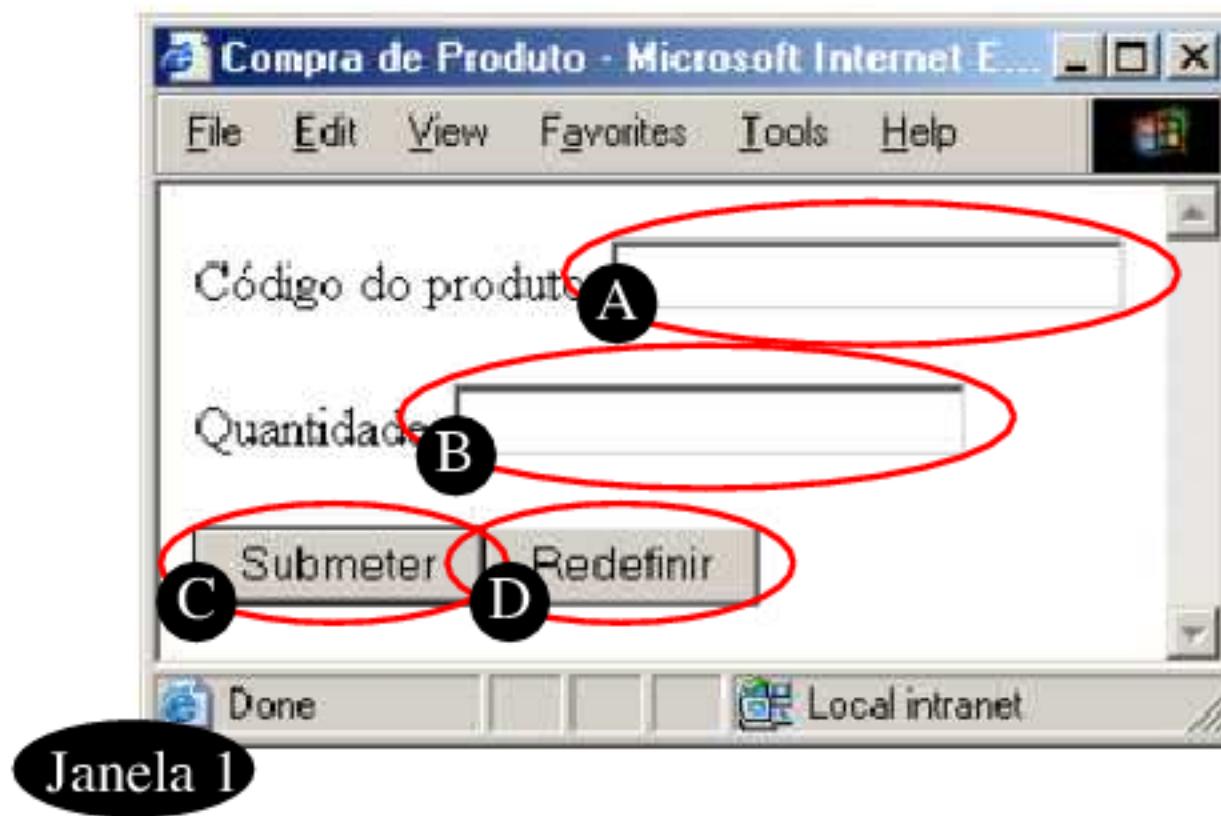
→ A descrição de casos de uso essenciais (análise) usualmente contém as seguintes informações:

- Nome;
- Resumo;
- Lista de atores envolvidos;
- Referências cruzadas (para requisitos ou outros casos de uso);
- Seqüência típica de eventos;
- Seqüências alternativas;

Descrição de casos de uso concretos

- ➡ A descrição de casos de uso concretos (projeto) refina a descrição de casos de uso essenciais incluindo:
 - Maquete de interface com o usuário com a identificação das *widgets*;
 - Utilização das *widgets* identificadas durante a descrição da seqüência típica de eventos;
 - Atualização da referência cruzada, descrevendo o caso de uso essencial que serviu como base;

Descrição de casos de uso concretos (exemplo)



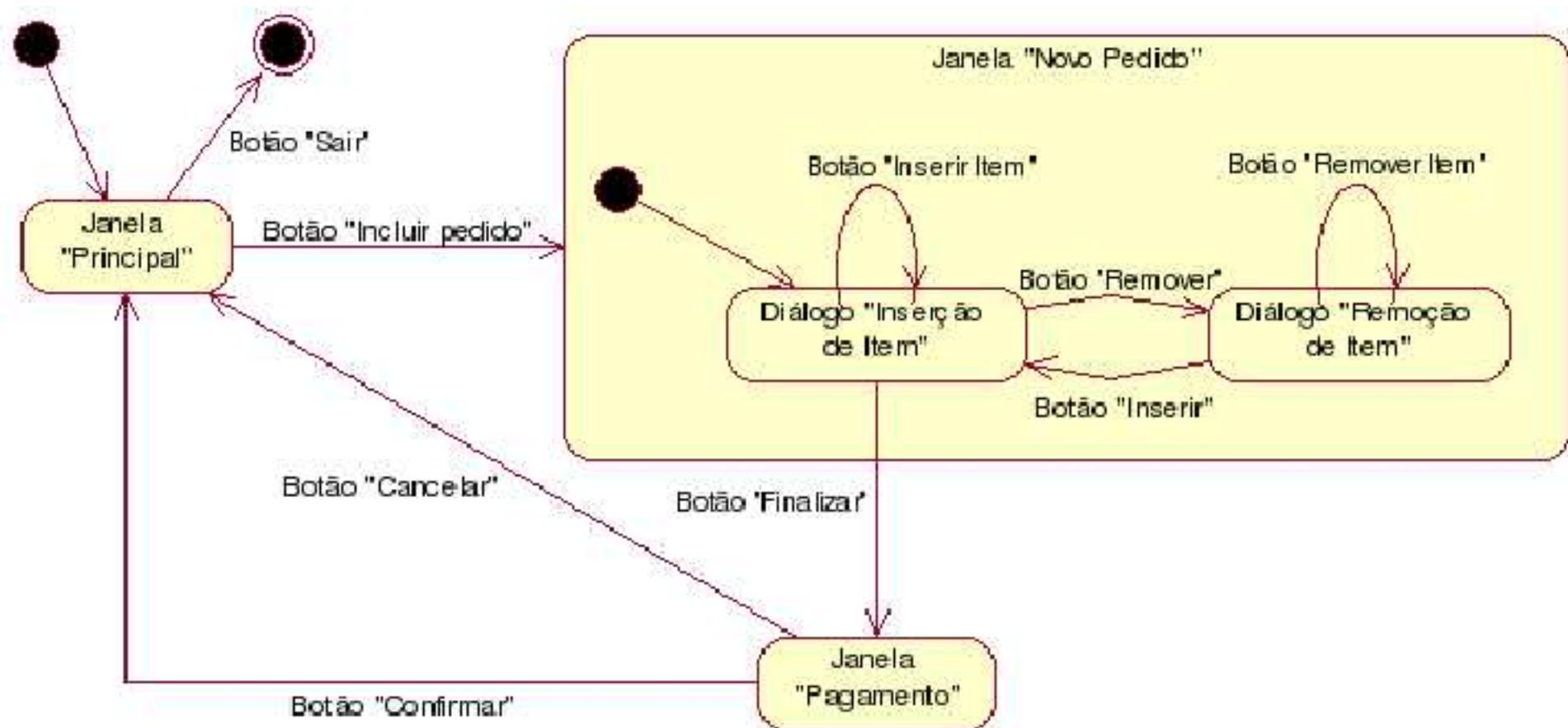
...

5 - O Cliente pressiona o botão "C" da "Janela 1" para enviar o pedido de compra...

Mapa de navegação

- Usualmente, o número de maquetes de um sistema é significativo;
- Para que seja possível visualizar o sistema como um todo é utilizado o **mapa de navegação**;
- O mapa de navegação também é conhecido como modelo de navegação;
- Esse modelo pode ser construído através do diagrama de transição de estados da UML, e utiliza a nomenclatura de janelas e campos definida nas maquetes;

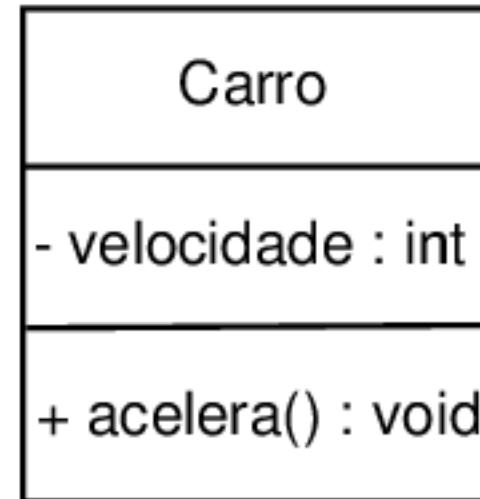
Mapa de navegação (exemplo)



Modelo de classes

→ As classes são descritas graficamente através de um retângulo, contendo três divisões:

- Nome;
- Atributos;
- Métodos;



Modelo de classes

→ A divisão de nome deve seguir a seguinte regra:

- Estereótipo (opcional);
- Nome da classe (em itálico para classe abstrata) segundo regra de nomenclatura definida pela linguagem de programação;
- Lista de propriedades entre chaves (opcional);

```
<<model>>
Carro
{ author = "Fulano", version = 1.0 }
```

Modelo de classes

→ A divisão de atributos deve seguir a seguinte regra:

- Um atributo por linha;
- Modificador de visibilidade (pode estar escondido):
 - Private (-);
 - Protected (#);
 - Public (+);
- Nome do atributo segundo regra de nomenclatura definida pela linguagem de programação;

```
- velocidade [1..1] : int = 3 { version = 1.5 }
    aceleracao : int
    COR : Color = Color.black { frozen }
```

Modelo de classes

- Multiplicidade entre colchetes (opcional para 1..1)
 - 0..1, 1..1, 0..*, 1..* ou intervalo determinado (eg.: 2..5);
- Tipo primitivo, definido pela linguagem de programação;
- Valor inicial (opcional);
- Lista de propriedades entre chaves (opcional);
- Atributos de classe são sublinhados;

```
- velocidade [1..1] : int = 3 { version = 1.5 }
    aceleracao : int
    COR : Color = Color.black { frozen }
```

Modelo de classes

→ A divisão de métodos deve seguir a seguinte regra:

- Um método por linha;
- Modificador de visibilidade (pode estar escondido):
 - Private (-);
 - Protected (#);
 - Public (+);
- Nome do método segundo regra de nomenclatura definida pela linguagem de programação;

```
+ acelera (aceleracao : int ) { version = 1.8 }
    getVelocidade () : int { query }
        getCor() : Color { abstract }
```

Modelo de classes

- Lista de parâmetros separados por vírgula:
 - Iniciadas pela classificação (in, out ou inout), onde a ausência representa in;
 - Nome do parâmetro;
 - Tipo primitivo;
 - Valor padrão;
- Lista de propriedades entre chaves (opcional);

```
+ acelera (aceleracao : int ) { version = 1.8 }
    getVelocidade () : int { query }
    getCor() : Color { abstract }
```

Modelo de classes

- Métodos de classe são sublinhados;
- Métodos abstratos podem estar em itálico ou receber a propriedade *abstract*;
- Métodos que não mudam o estado interno podem receber a propriedade *query*;
- Podem ser utilizadas propriedades para descrever a semântica de concorrência em métodos ou outras semânticas desejadas;

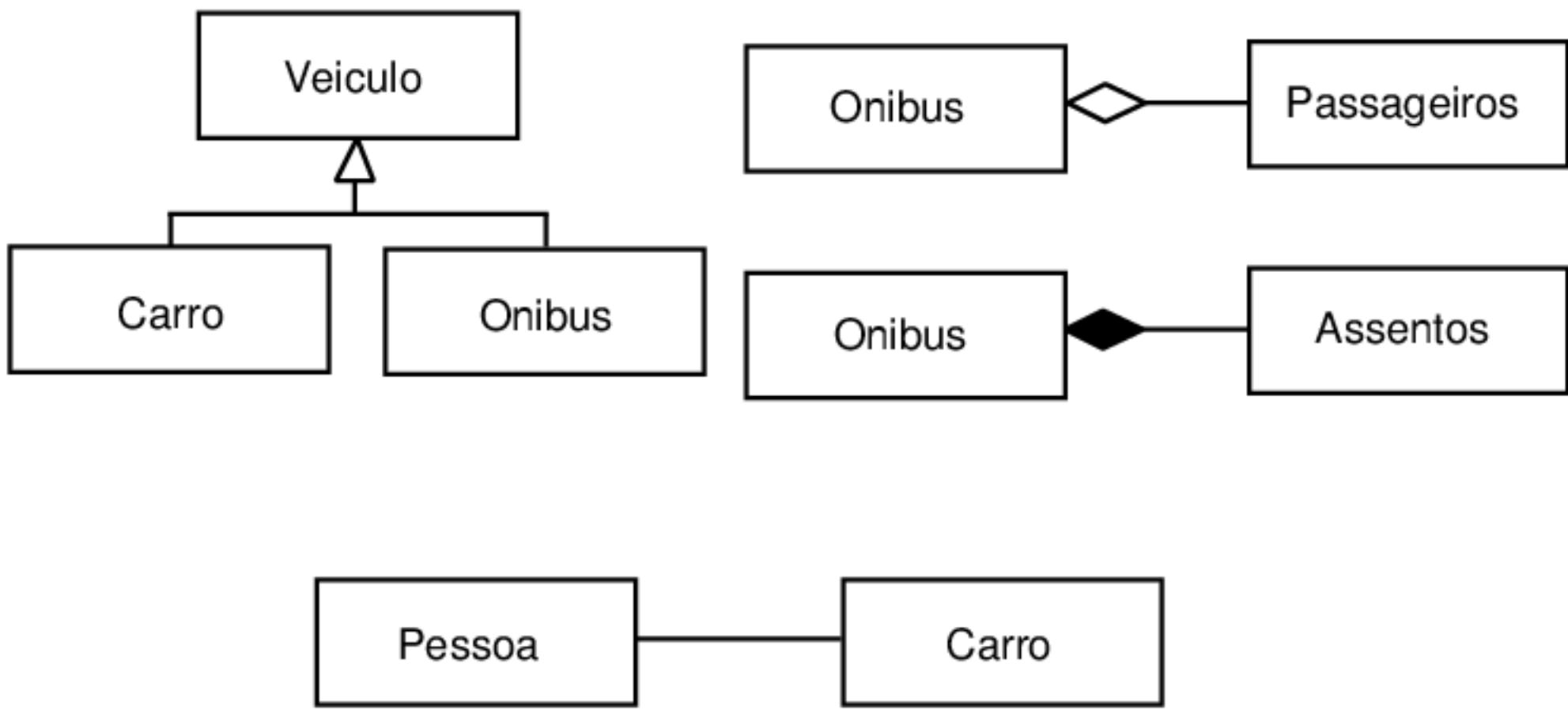
```
+ acelera (aceleracao : int ) { version = 1.8 }
    getVelocidade () : int { query }
    getCor() : Color { abstract }
```

Modelo de classes

→ As associações são descritas através de ligações contendo:

- Agregação
- Composição
- Generalização
- Associação

Modelo de classes



Modelo de seqüência

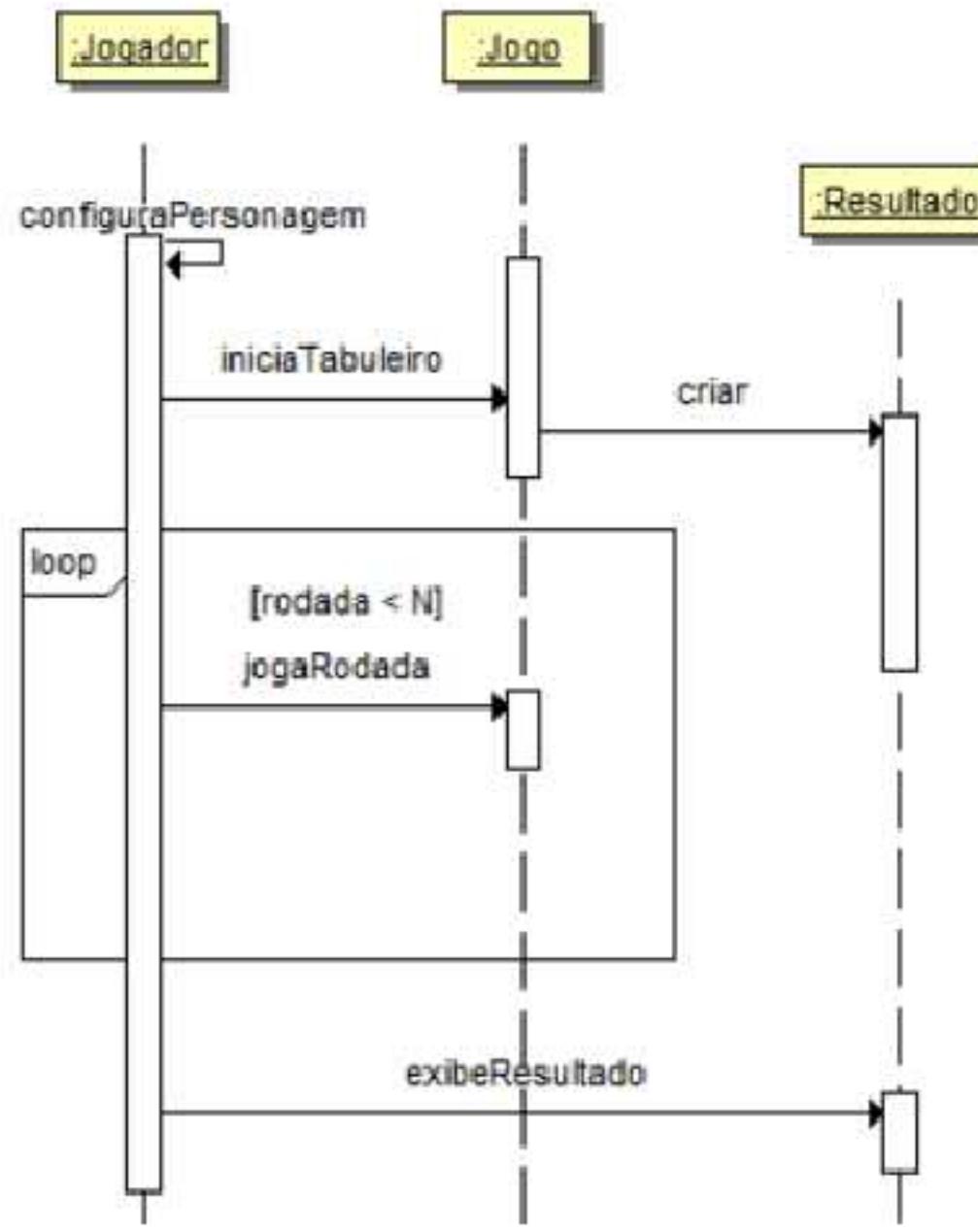
- ➡ Permite detalhar a interação entre objetos
- ➡ Descreve a seqüência de mensagens enviadas entre diversos objetos
- ➡ Normalmente são usados somente para detalhar comportamentos de alta complexidade
- ➡ Descrevem:
 - Objetos no topo
 - Linhas de vida dos objetos
 - Mensagens trocadas entre os objetos

Modelo de seqüênciā

→ Possibilidades adicionais:

- Criação de outros objetos
- Condição para o envio da mensagem
- Repetição no envio de uma mensagem para uma coleção de objetos
- Mensagem enviada para o próprio objeto

Modelo de seqüênciа



Modelo de transição de estado

- ➡ Modelam o comportamento de objetos, indicando os possíveis estados dos objetos e eventos que causam uma transição
- ➡ Acrescentam valor quando a classe exibe comportamento dinâmico interessante
- ➡ São utilizados por diversos métodos
 - Análise estruturada moderna
 - Análise essencial
 - Análise orientada a objetos

Modelo de transição de estado

→ Componentes:

- Estado: condição em que pode se encontrar um determinado elemento do sistema ao longo de sua existência; determina eventos, ações e condições possíveis no elemento
- Transição: representa uma mudança de estado, que pode ser automática ou gerada por eventos



Estado



Transição

Modelo de transição de estado

→ Um diagrama de estados tem no mínimo dois estados:

- Estado Inicial: um elemento recém criado no sistema se encontra neste estado
- Estado Final: estado final na cadeia de troca de estados do elemento; o elemento não poderá trocar de estado após atingir seu estado final; em um diagrama de estados podem existir diversos estados finais

Bibliografia

→ Básica

- Craig Larman, 2007, "Utilizando UML e Padrões", 3^a ed., Capítulos 15 e 16