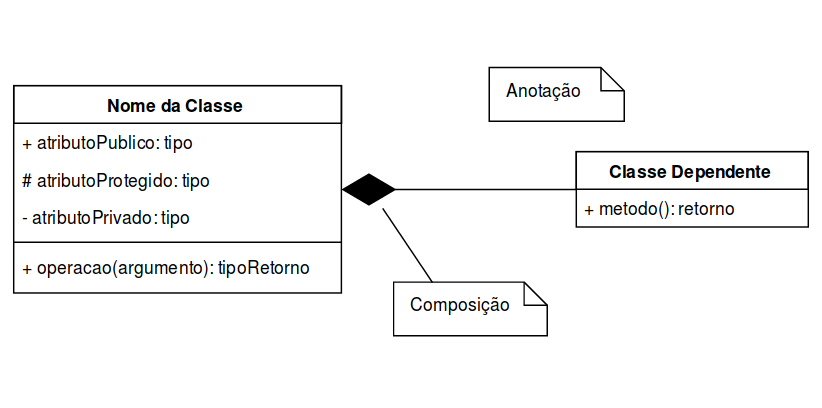
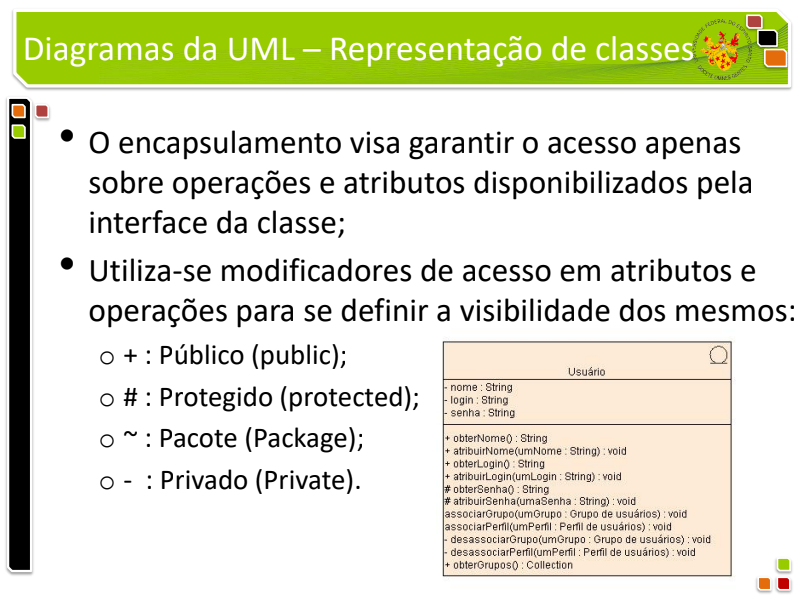
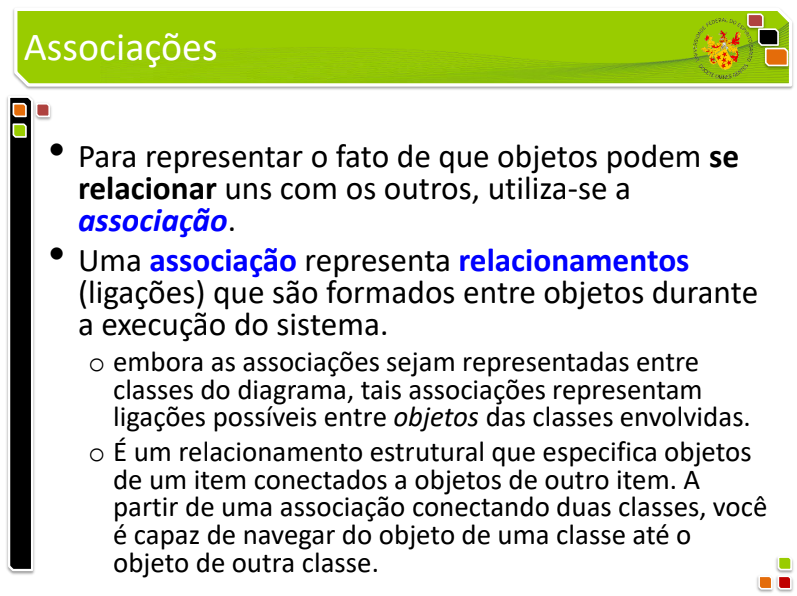
### **📘 1. Diagrama de Classes**

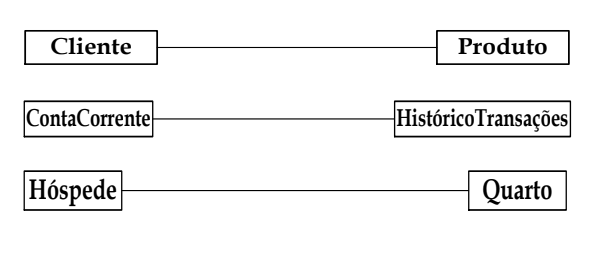
**Resumo:** Representa a estrutura estática do sistema, mostrando **classes**, seus **atributos**, **métodos** e os **relacionamentos** entre elas (herança, associação, agregação, composição).

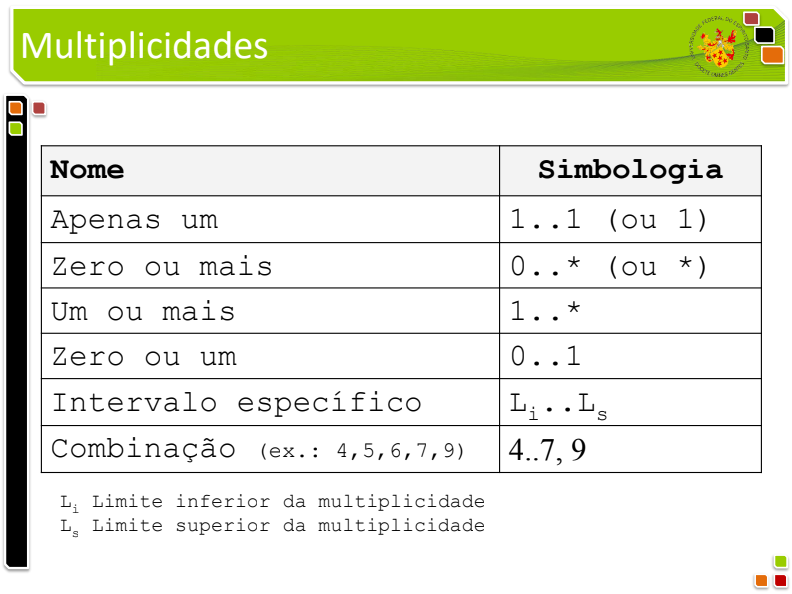
**Usado para:** Modelar a arquitetura do software e entender os objetos e suas interações.

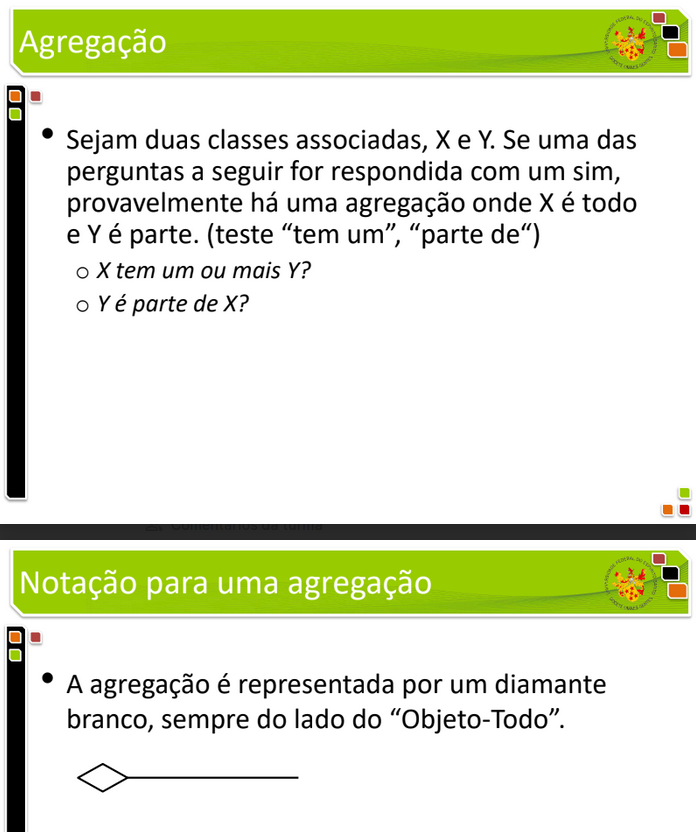


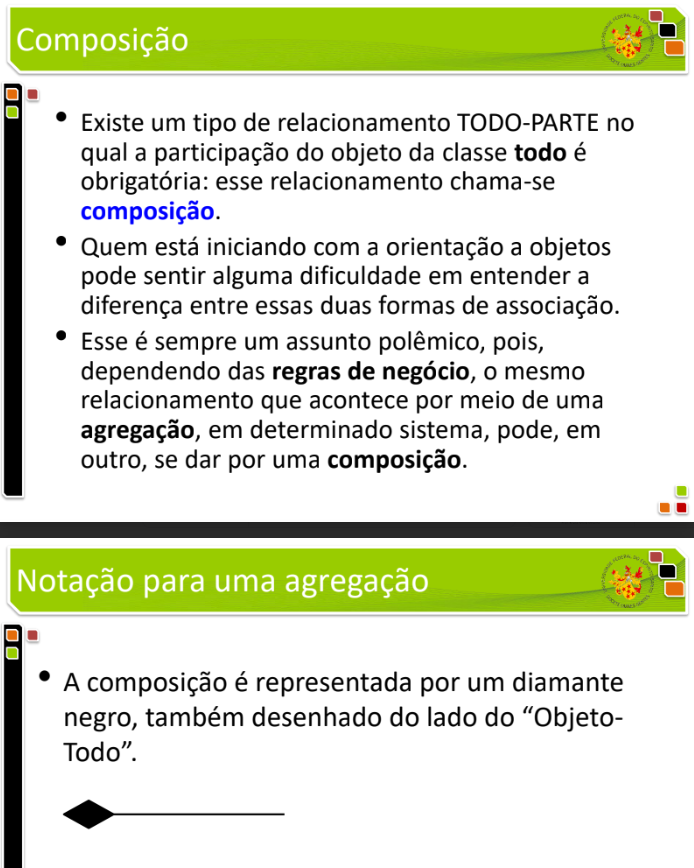


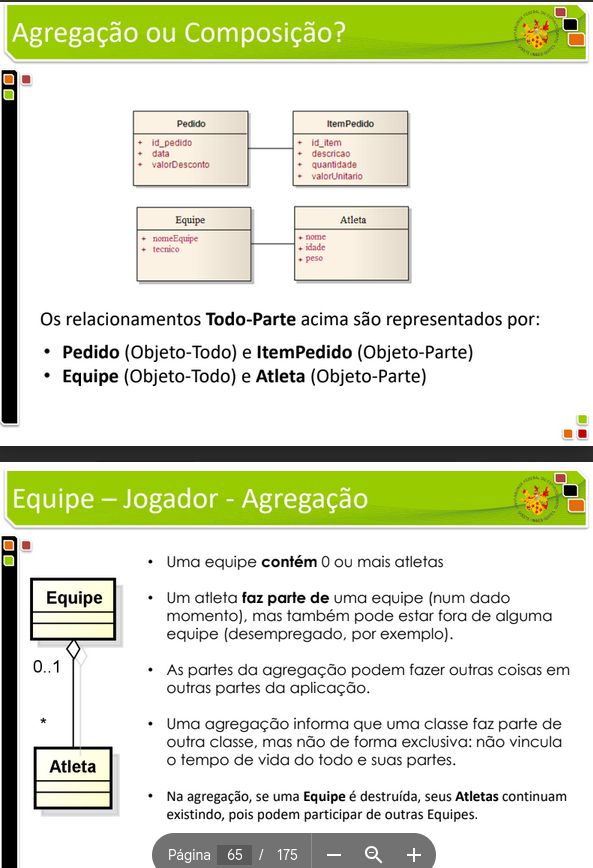


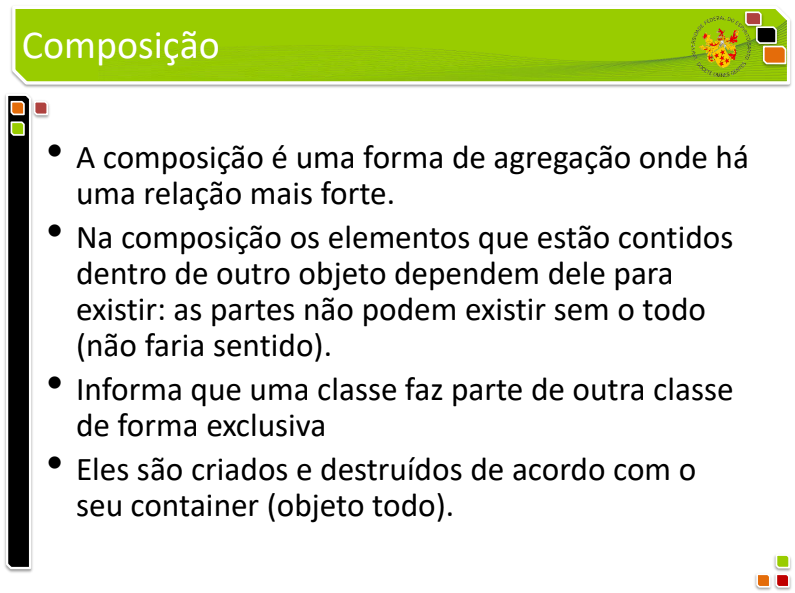


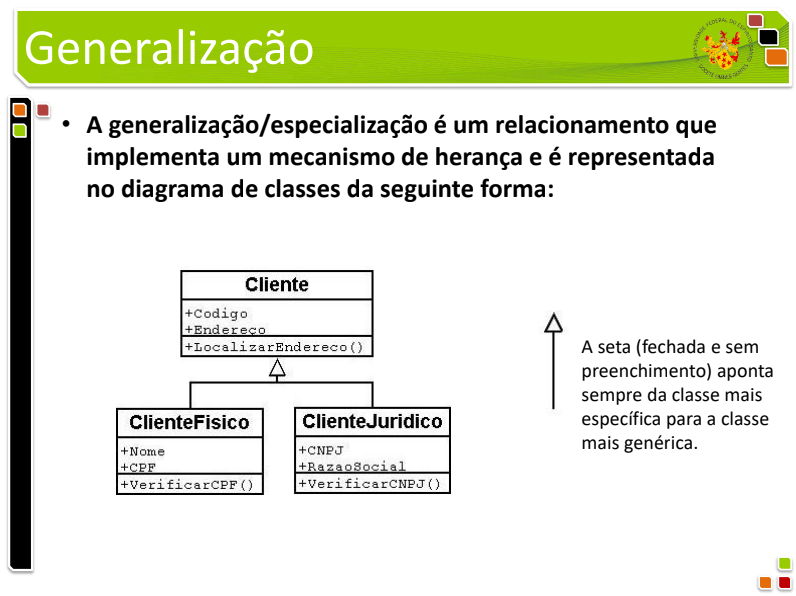


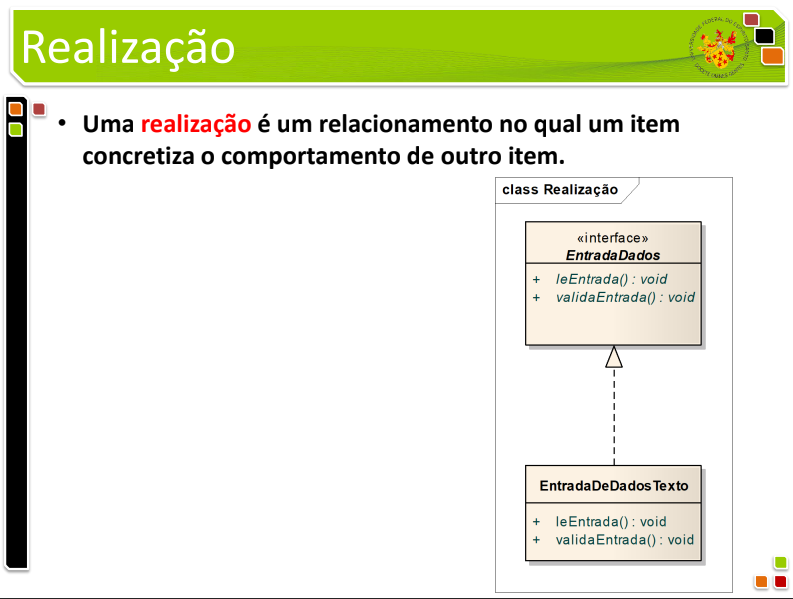


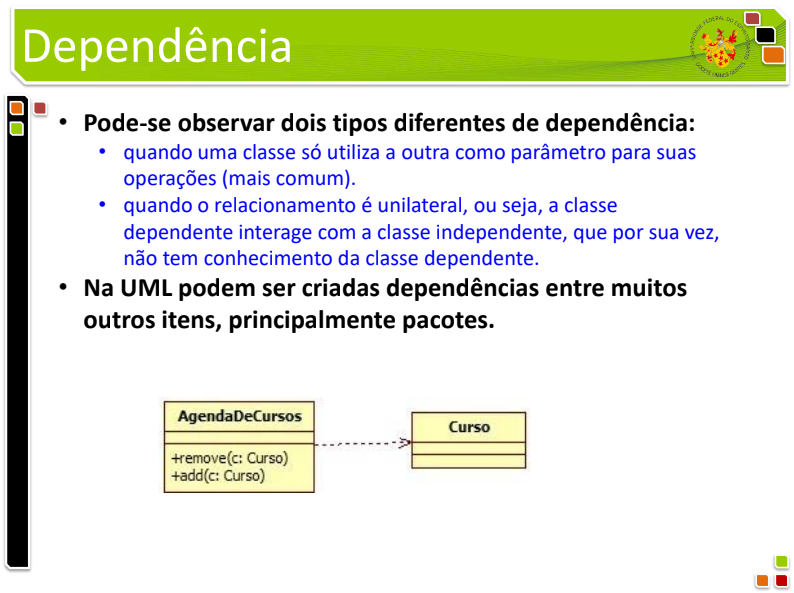












**O que é:** modelo **estático** da aplicação: classes, atributos, operações e **relacionamentos** (associação, dependência, generalização/herança, agregação, composição).

**Quando usar:** para estruturar o domínio/arquitetura, identificar responsabilidades e **multiplicidades** (1, 0..\*, etc.).

**Notação-chave:** visibilidade (+ público, − privado), tipos, relações com setas, estereótipos (<<interface>>), **agregação** (losango vazio), **composição** (losango preenchido).

**Erros comuns:** confundir agregação × composição; esquecer multiplicidade e navegabilidade; métodos demais (detalhe de implementação) cedo demais.

**Exemplo:** Em um sistema de e-commerce, podemos ter as classes Produto, Cliente e Pedido. Pedido possui uma lista de Produto (composição) e está associado a um Cliente. Multiplicidades indicam que um cliente pode ter vários pedidos e um pedido pode conter vários produtos.

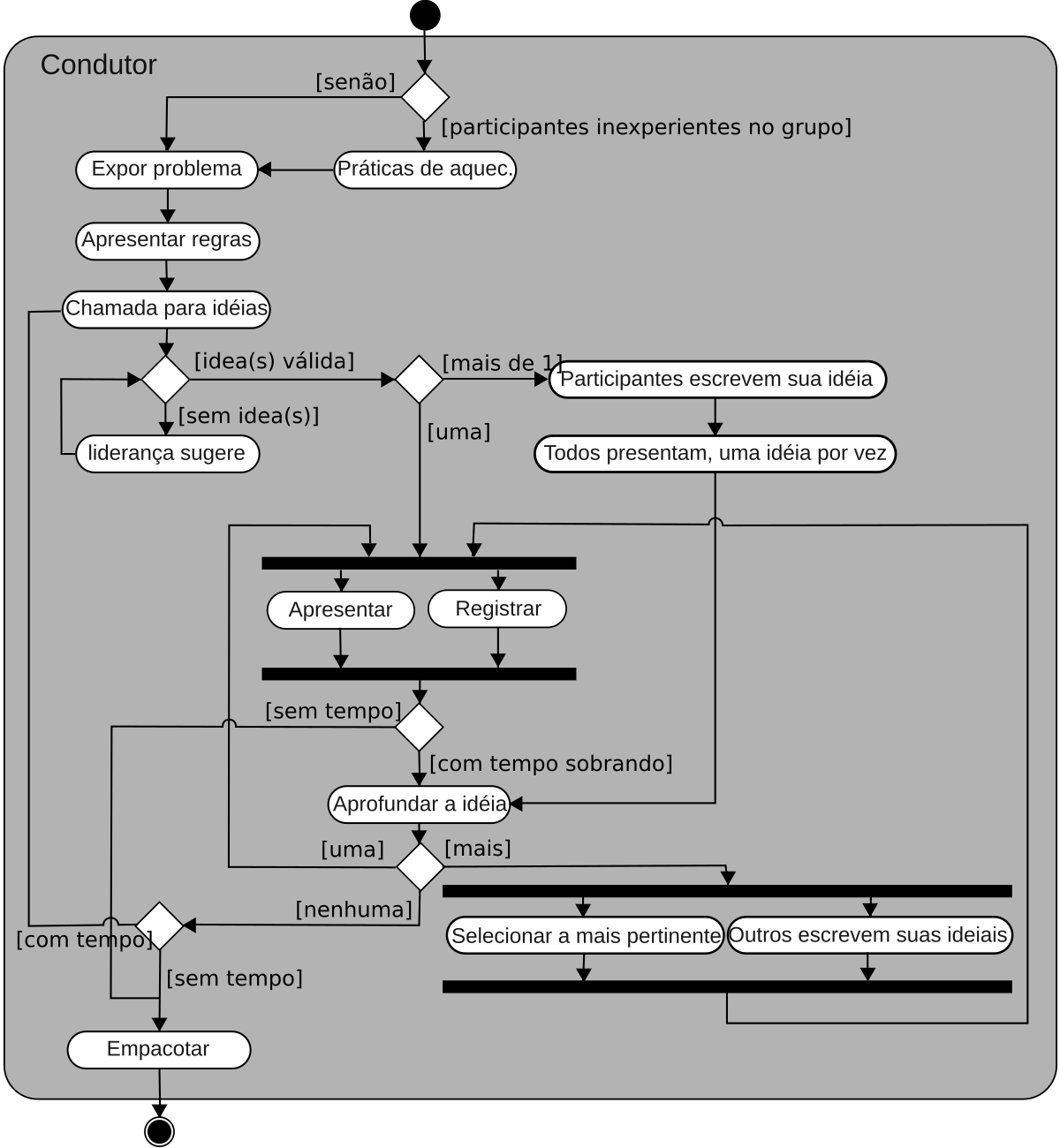
### 

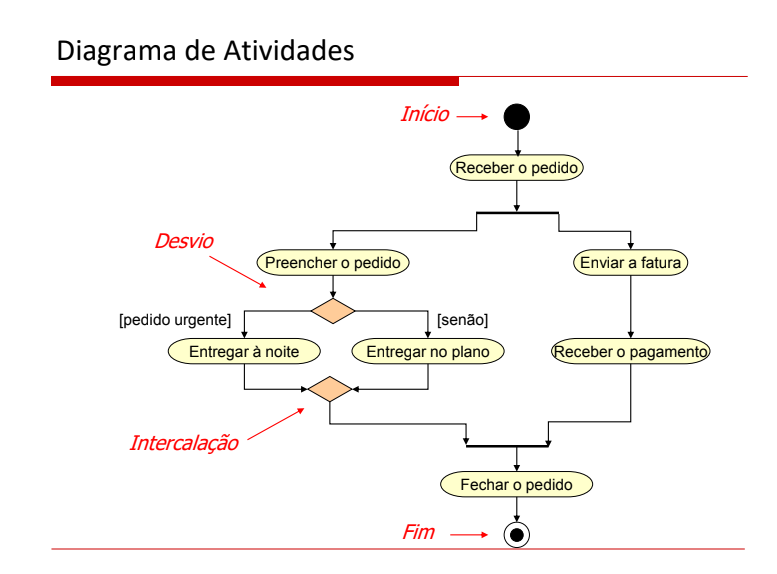
### 

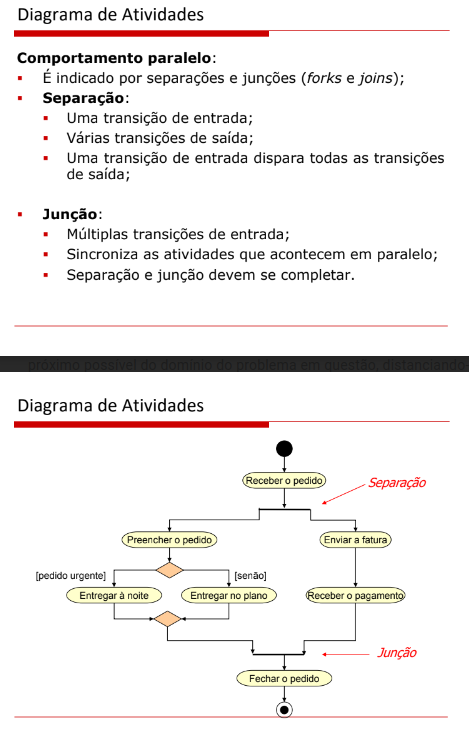
### **🔄 2. Diagrama de Atividades**

**Resumo:** Mostra o **fluxo de atividades** ou **processos** dentro de um sistema, com decisões, ramificações, paralelismos e ações.

**Usado para:** Representar fluxos de trabalho, como o processo de login ou compra online.







**O que é:** fluxo **procedimental** de atividades/ações, com **nós iniciais/finais**, **decisão/merge**, **fork/join** (paralelismo), **swimlanes** (responsáveis) e **objetos** fluindo.

**Quando usar:** modelar **processos/rotinas** (ex.: login, checkout); descrever regras de negócio com caminhos alternativos.

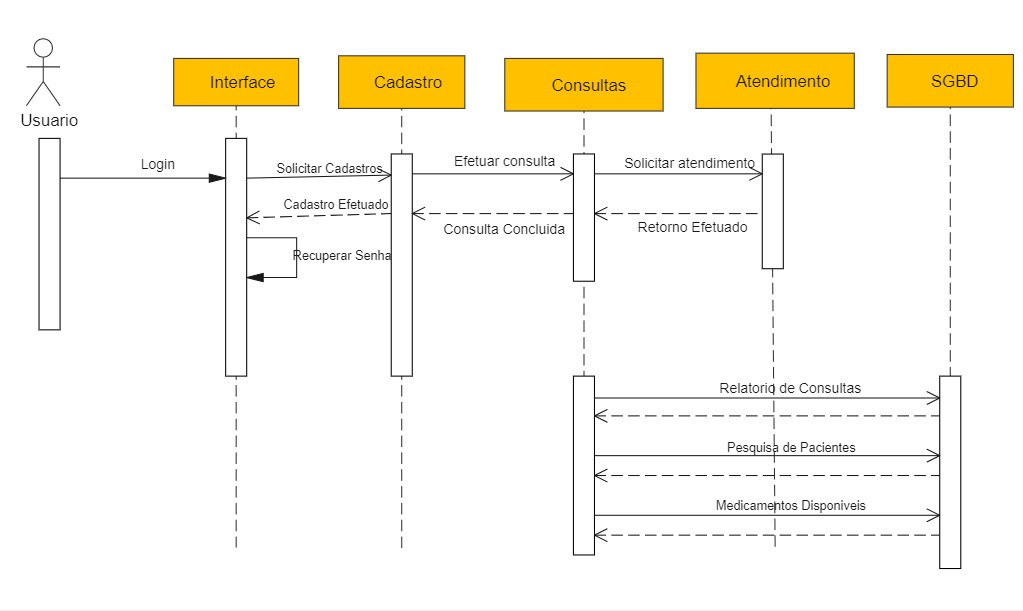
**Erros comuns:** misturar “o quê” (processo) com “quem/código”; esquecer guardas nas setas de decisão; paralelismo sem join.

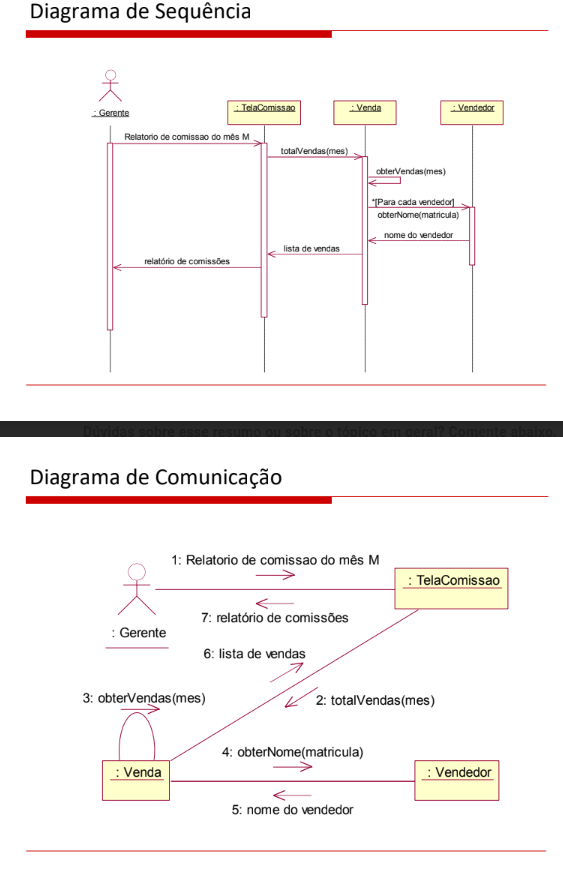
**Exemplo:** No processo de compra online: iniciar → selecionar produtos → adicionar ao carrinho → [decisão: cliente cadastrado?] → se sim, faz login; se não, realiza cadastro → escolher forma de pagamento → confirmar pedido → finalizar.

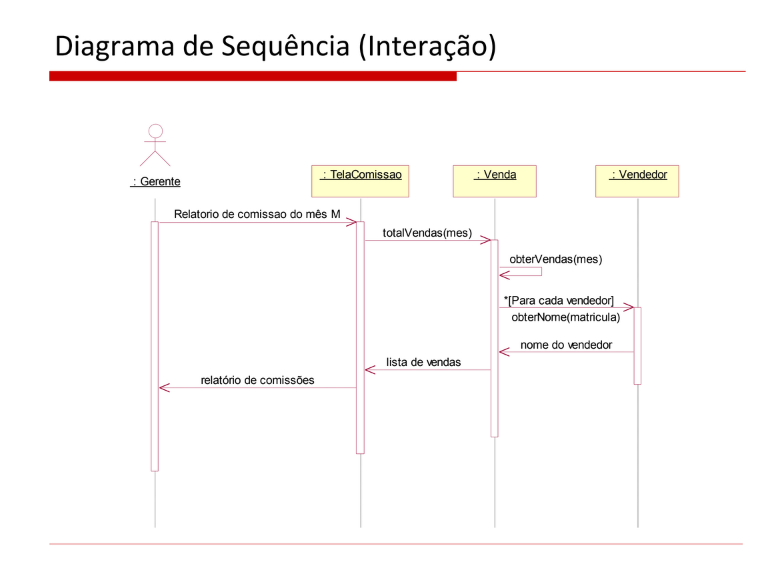
### **🕒 3. Diagrama de Sequência**

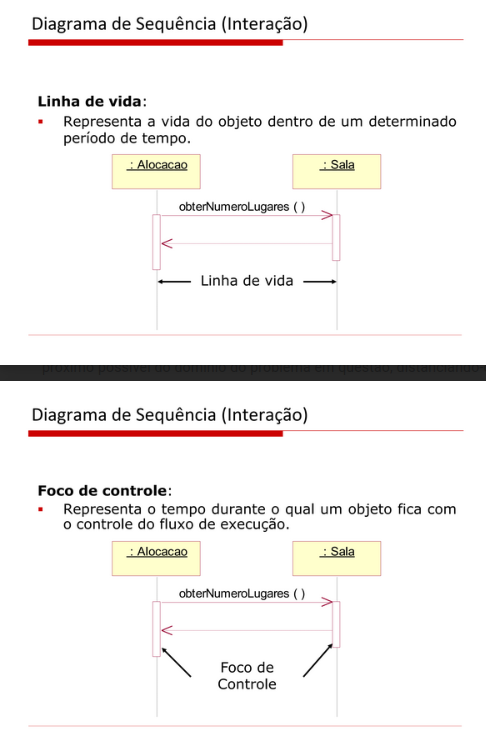
**Resumo:** Descreve como os **objetos interagem no tempo**, focando na troca de mensagens entre eles para realizar uma funcionalidade.

**Usado para:** Visualizar o comportamento de um caso de uso específico.









**O que é:** interação **no tempo** entre objetos/partes (lifelines, barras de ativação), **mensagens** síncronas/assíncronas, retornos e **fragmentos** (alt/opt/loop/par).

**Quando usar:** detalhar um **caso de uso** ou cenário crítico (incluindo exceções) e validar a colaboração entre componentes.

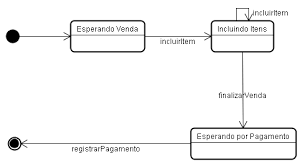
**Erros comuns:** mensagens sem ordem clara; não indicar quem inicia; esquecer condições/loops; “pular” camadas (UI → DB direto).

**Exemplo:** No caso de “realizar login”, o diagrama pode mostrar: Usuário envia dados de login para TelaLogin → esta envia para ControladorLogin → o controlador consulta BancoDeDados → retorna confirmação → a tela exibe mensagem de sucesso.

### **🔁 4. Diagrama de Estados**

**Resumo:** Mostra os **estados possíveis de um objeto** e as **transições** causadas por eventos.

**Usado para:** Modelar comportamentos dinâmicos, como o ciclo de vida de um pedido (Criado → Pago → Enviado → Entregue).



**O que é:** **ciclo de vida de um objeto**, com **estados**, **transições** rotuladas como evento [guarda] / ação, **entry/exit/do**, estado inicial/final e **subestados**.

**Quando usar:** objetos com comportamento **dirigido a eventos** (pedido, sessão, tarefa, dispositivo).

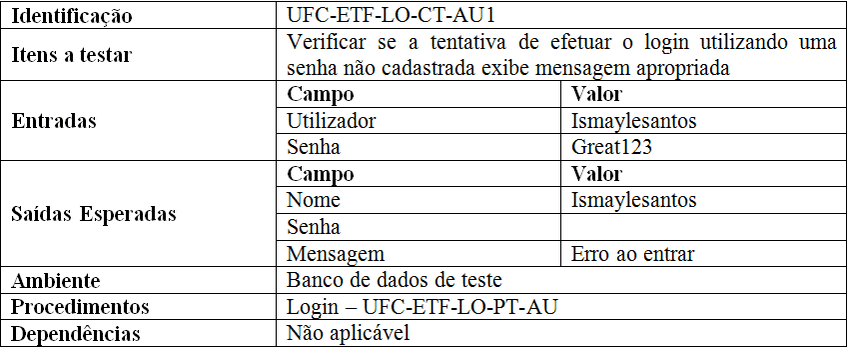
**Erros comuns:** usar para processo global (que é atividade); transições sem evento; não modelar condições de guarda.

**Exemplo:** Um pedido no e-commerce pode estar nos estados: Criado → (pagamento confirmado) → Pago → (produto enviado) → Enviado → (produto recebido) → Entregue. Caso o pagamento seja negado, há transição para Cancelado.

### **✅ 5. Casos de Teste**

**Resumo:** São descrições detalhadas que testam se uma funcionalidade do sistema está funcionando como esperado. Incluem **entrada, ação esperada e resultado esperado**.

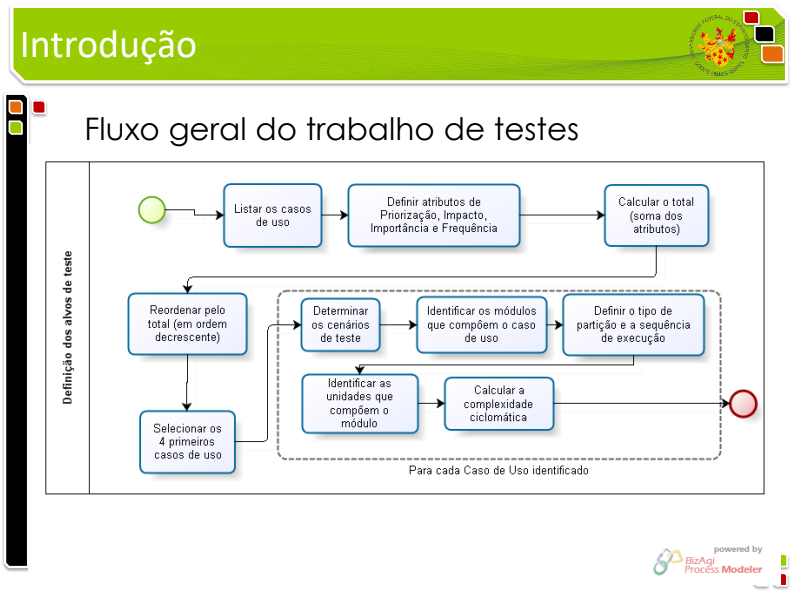
**Usado para:** Verificar se os requisitos do sistema foram corretamente implementados.

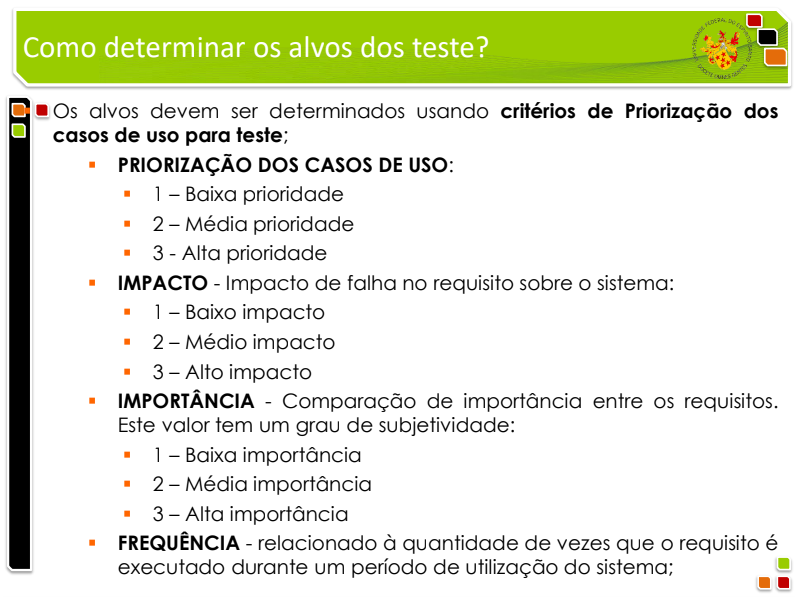


**O que são:** especificações verificáveis: **ID, título, objetivo, pré-condições, dados/steps, resultado esperado, pós-condições**, rastreio ao requisito.

**Técnicas úteis:** **particionamento por equivalência**, **análise de valores-limite**, **tabela de decisão**, **estado-transição**; níveis (unidade, integração, sistema, aceitação) e tipos (caixa-preta/caixa-branca).

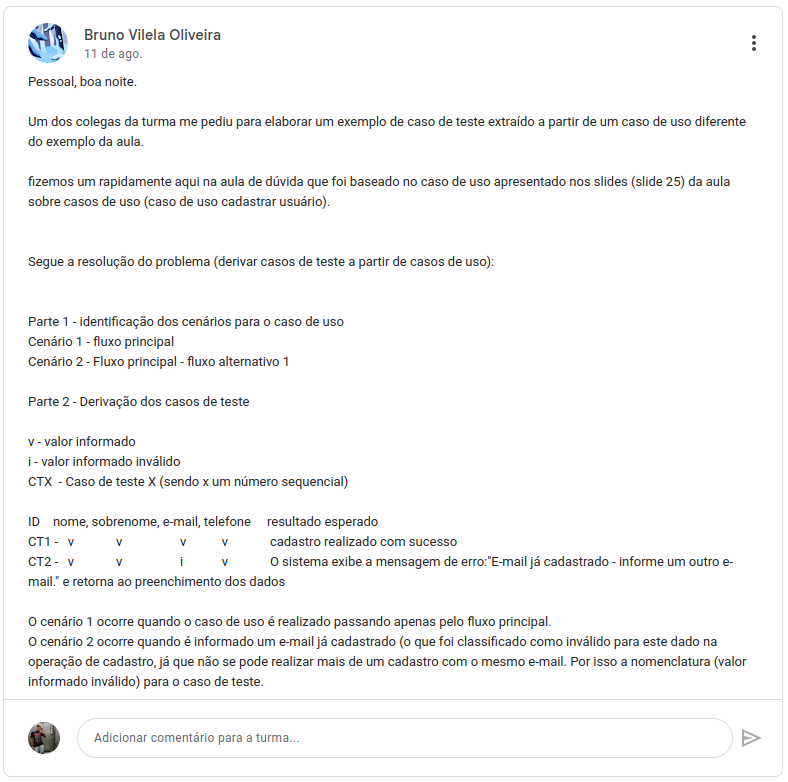
**Erros comuns:** casos sem oráculo (resultado esperado vago), falta de dados, ignorar cenários negativos e limites.





**Exemplo:**

* **ID:** CT-001
* **Objetivo:** Validar login com credenciais corretas.
* **Pré-condições:** Usuário já cadastrado.
* **Passos:** 1) Acessar página de login; 2) Inserir e-mail e senha válidos; 3) Clicar em “Entrar”.
* **Resultado esperado:** Sistema redireciona para página inicial do usuário.



Os **requisitos funcionais** descrevem **o que o sistema deve fazer**, ou seja, as **funcionalidades** e comportamentos que ele precisa oferecer para atender às necessidades do usuário ou do negócio.

* **Exemplos:**
  + "O sistema deve permitir que o usuário realize login com e-mail e senha."
  + "O sistema deve calcular o valor do frete com base no CEP informado."

Já os **requisitos não funcionais** definem **como o sistema deve se comportar** ou **as restrições/qualidades** que ele precisa atender, independentemente da funcionalidade.

* **Exemplos:**
  + "O sistema deve responder a qualquer solicitação em no máximo 2 segundos."
  + "O sistema deve estar disponível 99,9% do tempo."

📌 Resumindo:

* **Funcionais** → O **que** o sistema faz.
* **Não funcionais** → **Como** o sistema faz.

Acho que é isso!