CC6522
Modelagem de Software Orientado a Objetos

Modelagem de Sistemas
Introdução

centro universitário FEI

Prof. Dr. Ricardo Destro
2º Semestre de 2024

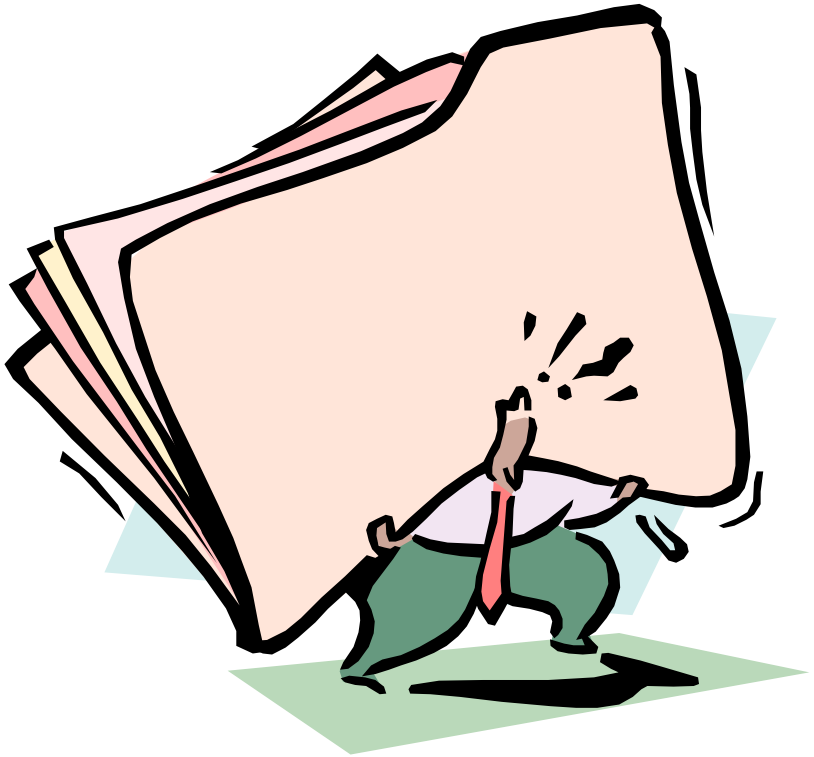
"A qualidade de um sistema é governada pela qualidade do processo utilizado para desenvolvê-lo e mantê-lo"

Watts Humphrey

centro universitário FEI

Peculiaridades do sofware

- Muitos pontos de vista.
  - Fabricantes / gestores / desenvolvedores
  - Clientes
- Surgimento de requisitos.
  - Vários clientes (externos e internos)
- Pequeno Time-to-Market.
- Carrega legado próprio e de outros.
- Interface com os fabricantes do Hw.
- Pode ser atualizado constantemente.



centro universitário FEI

Histórico de Metodologias

- Início Anos 70 - Programação Estruturada
  - Niklaus Wirth
- Fim Anos 70 - Projeto Estruturado
  - Constantine, Yourdon
- Início Anos 80 - Análise Estruturada
  - Yourdon/DeMarco, James Martim, Chris Gane
- Início Anos 90 - Análise Orientada por Objetos
  - Shlaer/Mellor, James Rumbaugh, Grady Booch, Jacobson.
- Fim Anos 90 - Maturidade em OO e Qualidade de Software - UML - Componentes

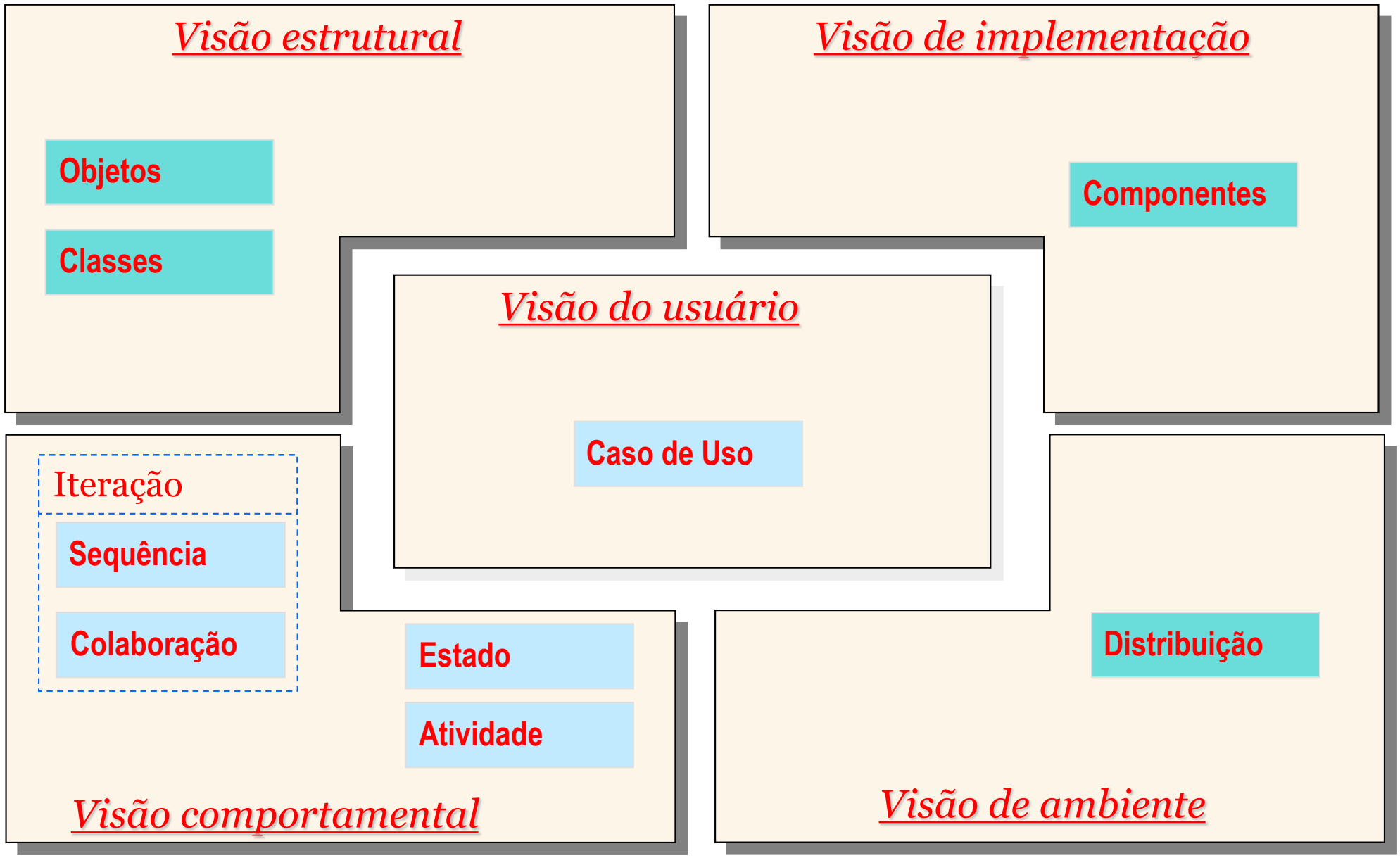
Diversas Metodologias OO no Mercado

- UML é uma **notação** padrão, e não pretende padronizar o **processo** intrínseco a cada método.
- Métodos diferentes podem utilizar a **mesma** linguagem para modelagem.
- UML é a Linguagem, o método é a frase.
- Vários metodologistas seguiram de perto este trabalho e anunciaram o suporte a esta notação, como Steve Mellor, Bertrand Meyer, Rebeca Wirfs, James Martin, e outros.

Diagramas

- Um Diagrama é uma visão do modelo
  - Apresentado da perspectiva de um patrocinador em particular
  - Fornece uma representação parcial do sistema
  - É semanticamente consistente com outras visões
- Na UML, há nove diagramas padrão:
  - Visão estática: casos de usos, classes, objetos, componentes, distribuição
  - Visão dinâmica: seqüência, colaboração, máquinas de estados, atividades

Modelos, Visões, Diagramas



Dicas

- Nem todos os Diagramas necessitam ser utilizados;
- Evite Diagramas estranhos ou redundantes;
- Utilize apenas informações coerentes para os propósitos da Modelagem;
- Evite a poluição nos Diagramas;
- Não simplifique demais os Diagramas;
- Faça um balanceamento dos Diagramas Comportamentais, Estruturais e Funcionais do Sistema;
- Utilize nomes significativos nos Diagramas ;
- Use Ferramentas CASE para desenhar os Diagramas.

centro universitário FEI

CC6522
Modelagem de Software Orientado a Objetos

Requisitos

lembretes rápidos.... lembrem-se do semestre passado em Eng. Software!

centro universitário FEI

Requisito : uma condição ou capacidade que um sistema deve apresentar

- Funcionalidades
- Qualidades

Necessária ao usuário do sistema, para resolver um problema e alcançar um objetivo

Aquilo que deve ser definido antes de começar a construir um sistema

centro universitário FEI

Definição dos Requisitos

centro universitário FEI

Mudança nos Requisitos

Re-projeto

centro universitário FEI

Entrega do Sistema

Requisitos

- Funcionais

  Descrevem as funcionalidades que se espera que o sistema disponibilize, de uma forma completa e consistente.

  Relacionados a Entradas, Funções, Saídas, Atores.

- Não-funcionais

  Referem-se às restrições nas quais o sistema deve operar ou propriedades emergentes do sistema (como viabilidade ou tempos de resposta).

  Tipos

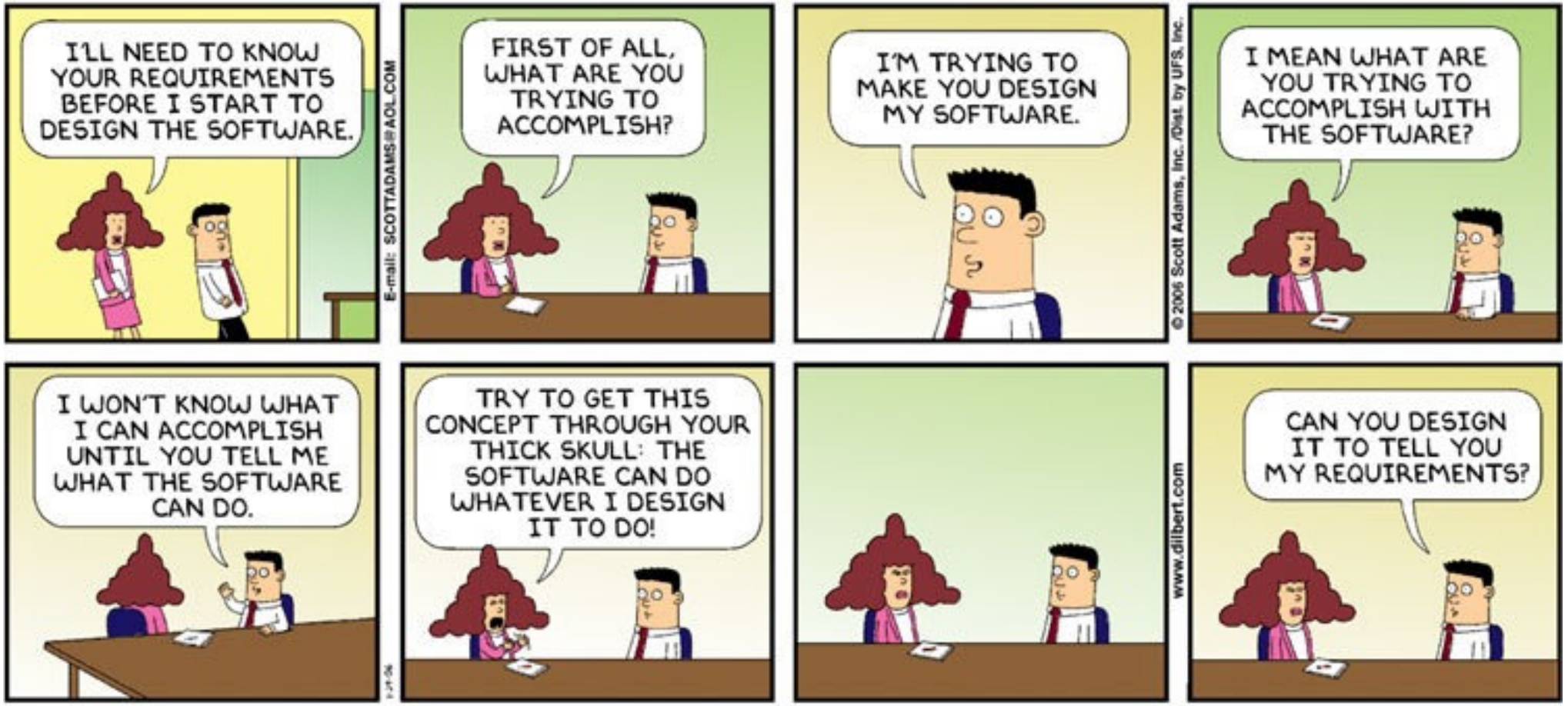
  - Produto (Eficiência, Portabilidade, Segurança, etc.);
  - Organizacionais (Padrões, Entrega, etc.);
  - Externos (Aspectos Éticos, Legais, etc.).

Requisitos não funcionais

Apenas alguns exemplos / lembretes ;)

- **Operacional**
  - ambiente operacional
  - condições do usuário
  - sistemas relacionados
- **Segurança**
  - Confidencialidade
  - Integridade
  - Disponibilidade
- **Legal**
  - Leis
  - Regulamentações
  - normas existentes
- **Suporte**
  - Capacidade manter o sistema atualizado
- **Usabilidade**
  - facilidade de uso pelos usuários
- **Confiança**
  - frequência e resistência a falhas
  - capacidade de recuperação
  - Predibilidade
  - Precisão
- **Desempenho**
  - Capacidade
  - taxas em relação ao tempo (tempo de resposta, disponibilidade)
  - Precisão
  - Uso de memória
- **Aparência**
  - Visual
  - Design gráfico

centro universitário FEI



Casos de uso

Um lembrete nem tão rápido assim...
Lembrem-se do semestre passado....
vamos aprofundar um pouco....
e rever a aplicação...

Use Case e Atores

- Um *use case* (caso de uso) é uma seqüência de ações executada pelo sistema que gera um resultado de valor observável para um ator particular;

- Um ator é alguém ou alguma coisa fora do sistema que interage com o sistema;

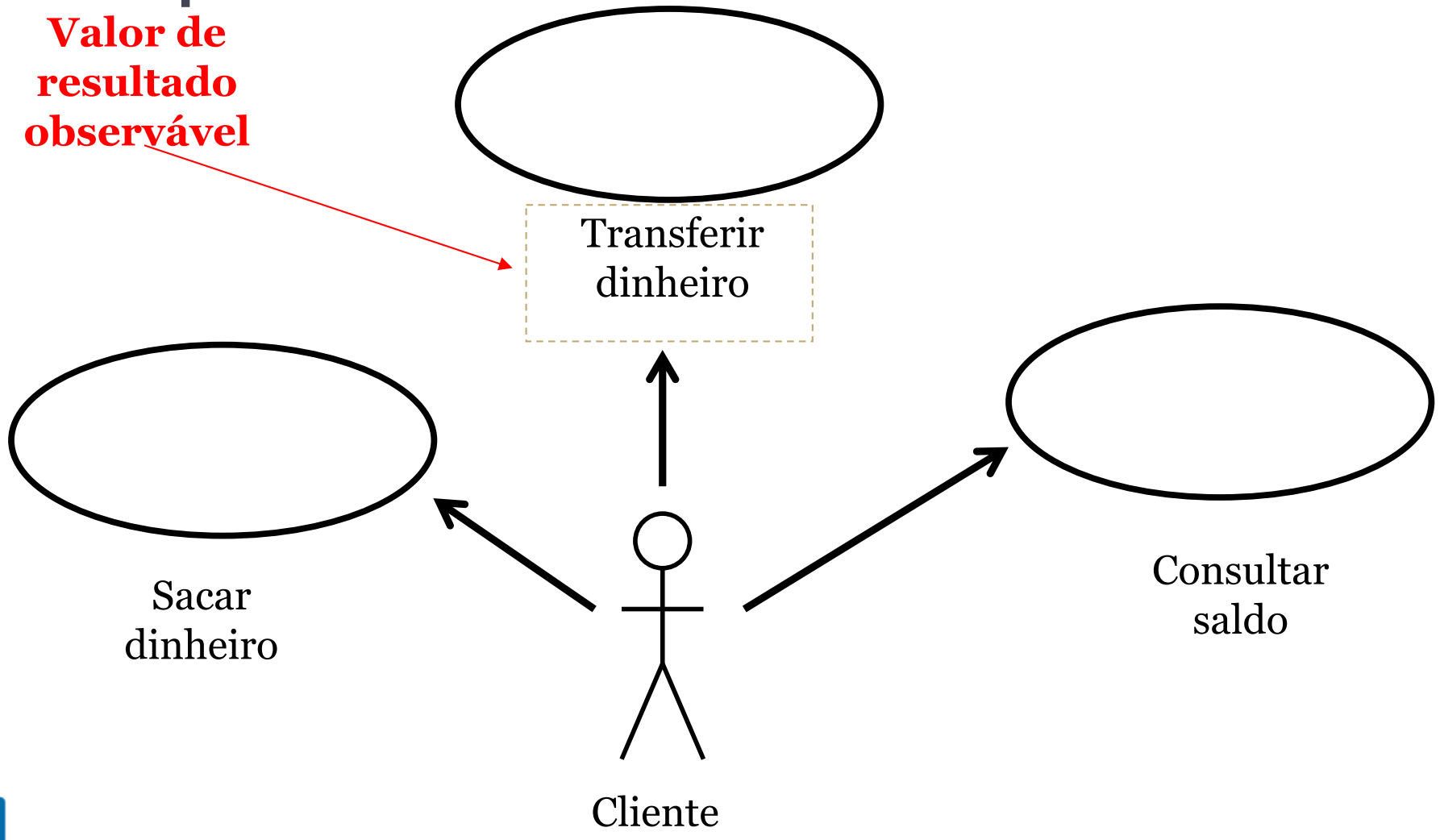
Use Case e Ator

- A descrição de um *use case* define o que o sistema faz quando o *use case* é realizado;
- A funcionalidade do sistema é definida por um conjunto de *use cases*, cada um representando um fluxo de eventos específico;

Exemplo de *Use Case* e Ator

- Um cliente de um banco pode usar um caixa automático (CA) para sacar dinheiro, transferir dinheiro ou consultar o saldo da conta
- Ator: Cliente
- *Use cases*: Sacar dinheiro, transferir dinheiro e consultar saldo

Exemplo de *Use Case* e Ator



Identificando *Use Cases*

- Em geral é difícil decidir se um conjunto de interações usuário-sistema é um ou são vários *use cases;*

- Por exemplo, seriam *use cases:*
  - Inserir cartão em um ATM?
  - Entrar com a senha?
  - Receber o cartão de volta?
- *Use cases* têm que representar valor para o ator;

- Portanto, transferir dinheiro, sacar, depositar e consultar saldo seriam *use cases;*

centro universitário FEI

Identificando *Use Cases*

- Determinam-se a partir de interações ou agrupamento de seqüência de ações com o sistema que resultam valores para atores;
- Uma alternativa seria que um *use case* satisfaz um objetivo particular de um ator que o sistema deve prover;
- A razão para agrupar certas funcionalidades e chamá-las de *use cases* é facilitar seu gerenciamento em conjunto durante todo o ciclo de desenvolvimento;

Evolução de *Use Cases*

- Inicialmente, *use cases* podem ser tão simples que apenas um esboço sobre seu funcionamento é suficiente;
- Porém, com a sedimentação da modelagem, uma descrição detalhada de seu fluxo de eventos faz-se necessária;
- O fluxo de eventos deve ser refinado até que todos os *stakeholders* envolvidos estejam de acordo com a descrição;

Explorando Reuso em *Use Cases*

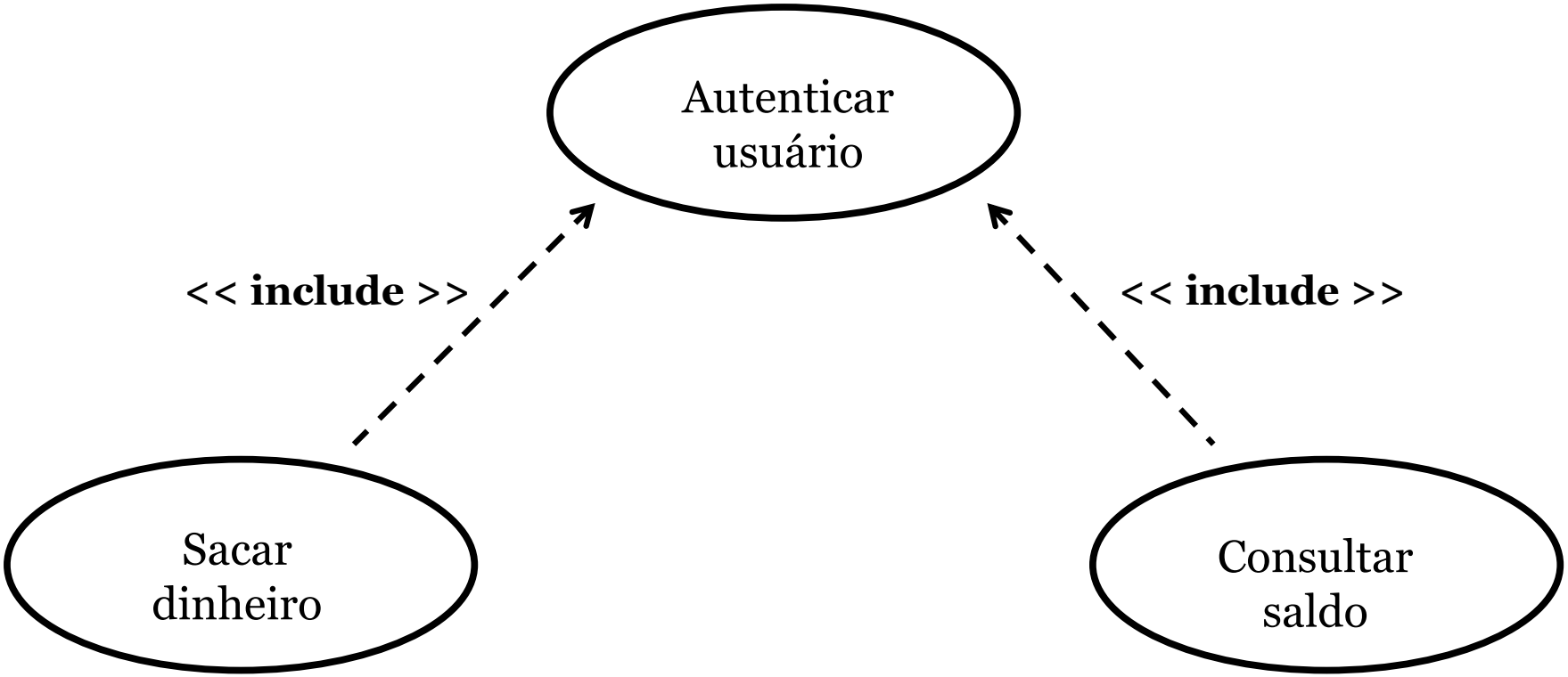
- Se o comportamento é comum a mais de dois *use cases* ou se ele forma uma parte independente
- Então pode-se modelá-lo como um *use case* em si para ser reusado por outros
- Existem três maneiras de reusar:
  - Inclusão
  - Extensão
  - Generalização/Especialização

Inclusão

- Vários use cases podem incluir um texto de fluxo de eventos comum organizado como um use case;
- Equivalente a fatoração feita em programação através de sub-programas;
- Por exemplo, tanto “Sacar dinheiro” quanto “Consultar saldo” necessitam da senha;
- Poder-se-ia criar “Autenticar usuário” e incluí-lo nos use cases anteriores;
  Mas atenção, não se deve criar use cases mínimos.
- O que importa é que um use case tenha algum valor para ator particular

centro universitário FEI

Inclusão


Autenticar usuário
<< include >>
<< include >>
Sacar dinheiro
Consultar saldo

Inclusão

- Descrição de Consultar saldo

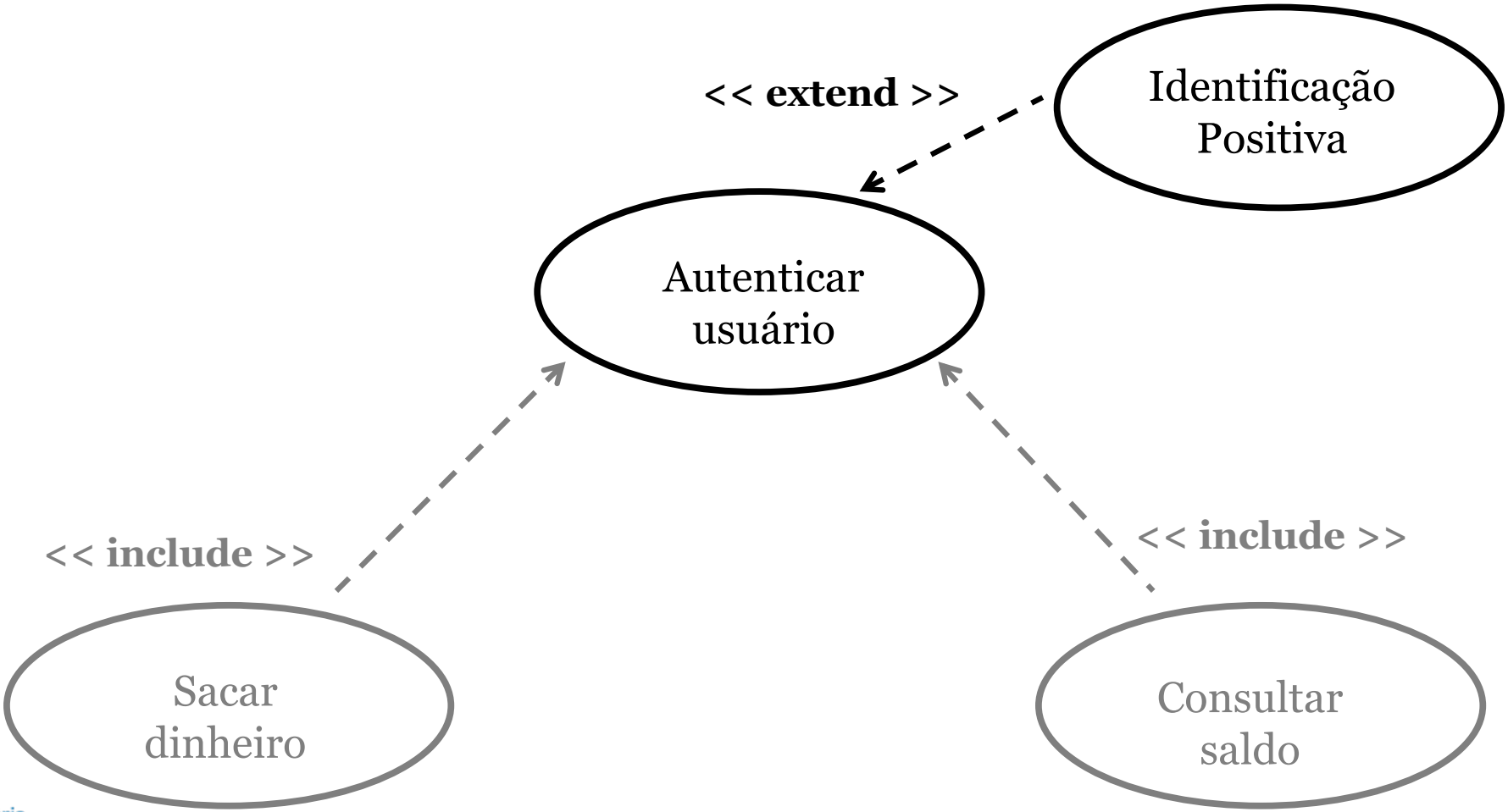
  Fluxo de Eventos Principal:

  - include (Autenticar usuário). Sistema pede a Cliente que selecione tipo de conta (corrente, etc). ...

Extensão

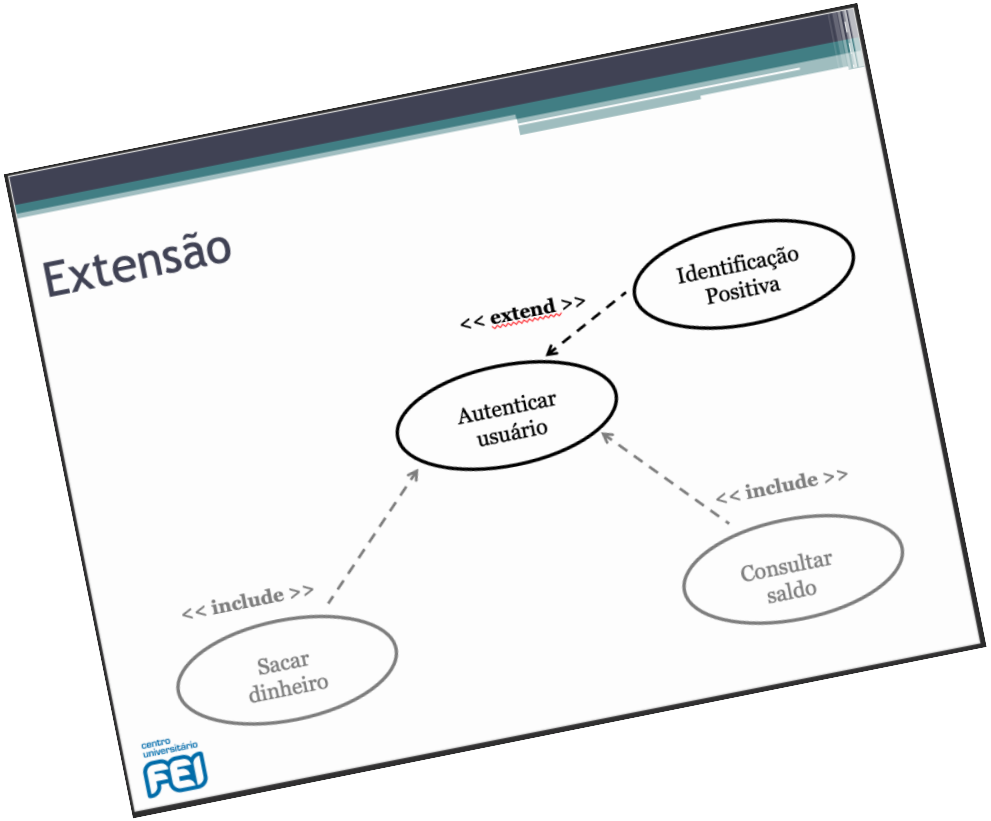
- Um *use case* pode ser estendido por outro (estender funcionalidade);
- A extensão ocorre em pontos específicos chamados de pontos de extensão;
- Extensão se dá pela inclusão de texto adicional (fluxo de eventos), nos pontos de extensão sob condições particulares;
- Pode ser usada para simplificar fluxos de eventos complexos, representar comportamentos opcionais e lidar com exceções;

Extensão



centro universitário FEI

Extensão



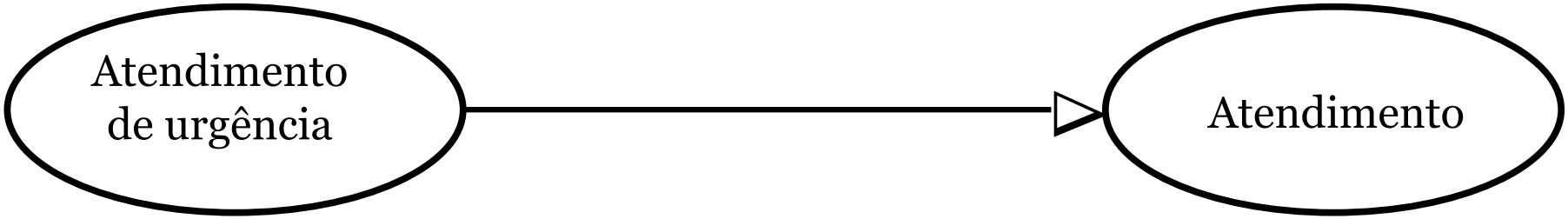
Conseguem encontrar um outro exemplo de extensão neste diagrama?

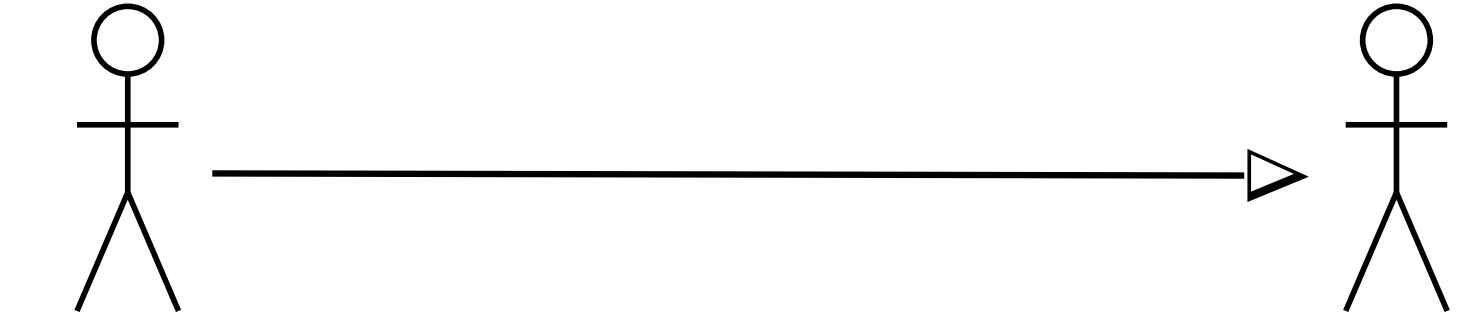
centro universitário FEI

Especialização

- Um *use case* pode especializar um outro através da adição ou refinamento do fluxo de eventos original
- Especialização fornece uma maneira de modelar o comportamento de estruturas de aplicação em comum
- Especialização é o caminho oposto ao de generalização

Especialização





centro universitário FEI

Fluxo de Eventos

- A parte mais importante de um *use case* na atividade de requisitos é seu fluxo de eventos;
- O fluxo de eventos define a seqüência de ações entre o ator e o sistema;
- Geralmente escrito em linguagem natural, com um uso preciso de termos definidos no glossário de acordo com o domínio do problema;
- Também pode ser descrito usando texto formal (pré e pós-condições) ou pseudo-código;

centro universitário FEI

Exemplo de Fluxo de Eventos

- Um esboço inicial sobre Sacar dinheiro seria
  1. O *use case* inicia quando o Cliente insere um cartão no CA. Sistema lê e valida informação do cartão
  2. Sistema pede a senha. Cliente entra com a senha. Sistema valida a senha.
  3. Sistema pede seleção do serviço. Cliente escolhe "Sacar dinheiro"
  4. Sistema pede a quantia a sacar. Cliente informa.
  5. Sistema pede seleção da conta (corrente, etc). Cliente informa.
  6. Sistema comunica com a rede para validar a conta, senha e o valor a sacar.
  7. Sistema pede remoção do cartão. Cliente remove.
  8. Sistema entrega quantia solicitada.

Fluxo de Eventos

- Quando se tenta descrever o que o sistema faz através de fluxos de eventos completos, surgem vários caminhos possíveis
- Existem caminhos alternativos e casos diferentes a considerar, e efeitos ou valores diferentes são produzidos
- Coletivamente, o fluxo de eventos do *use case* eventualmente descreve todos esses caminhos possíveis

Sub-fluxos de Eventos

- Pode-se ver um fluxo de eventos como vários sub-fluxos de eventos;
- Um sub-fluxo é tido como o principal e os demais como alternativos;
- O interessante dessa abordagem é o reuso de fluxos de eventos de certos *use cases* por outros *use cases;*

Exemplo de Sub-fluxos

- Seja o *use case* Validar usuário

  Fluxo principal:

  - O *use case* inicia quando o sistema pede ao Cliente a senha. Cliente entra com senha. Sistema verifica se a senha é válida. Se a senha é válida, sistema confirma e termina o *use case*.

  Fluxo excepcional:

  - Cliente pode cancelar a transação a qualquer momento pressionando a tecla ESC, reiniciando o *use case*. Nenhuma modificação é feita na conta do Cliente.

  Fluxo excepcional:

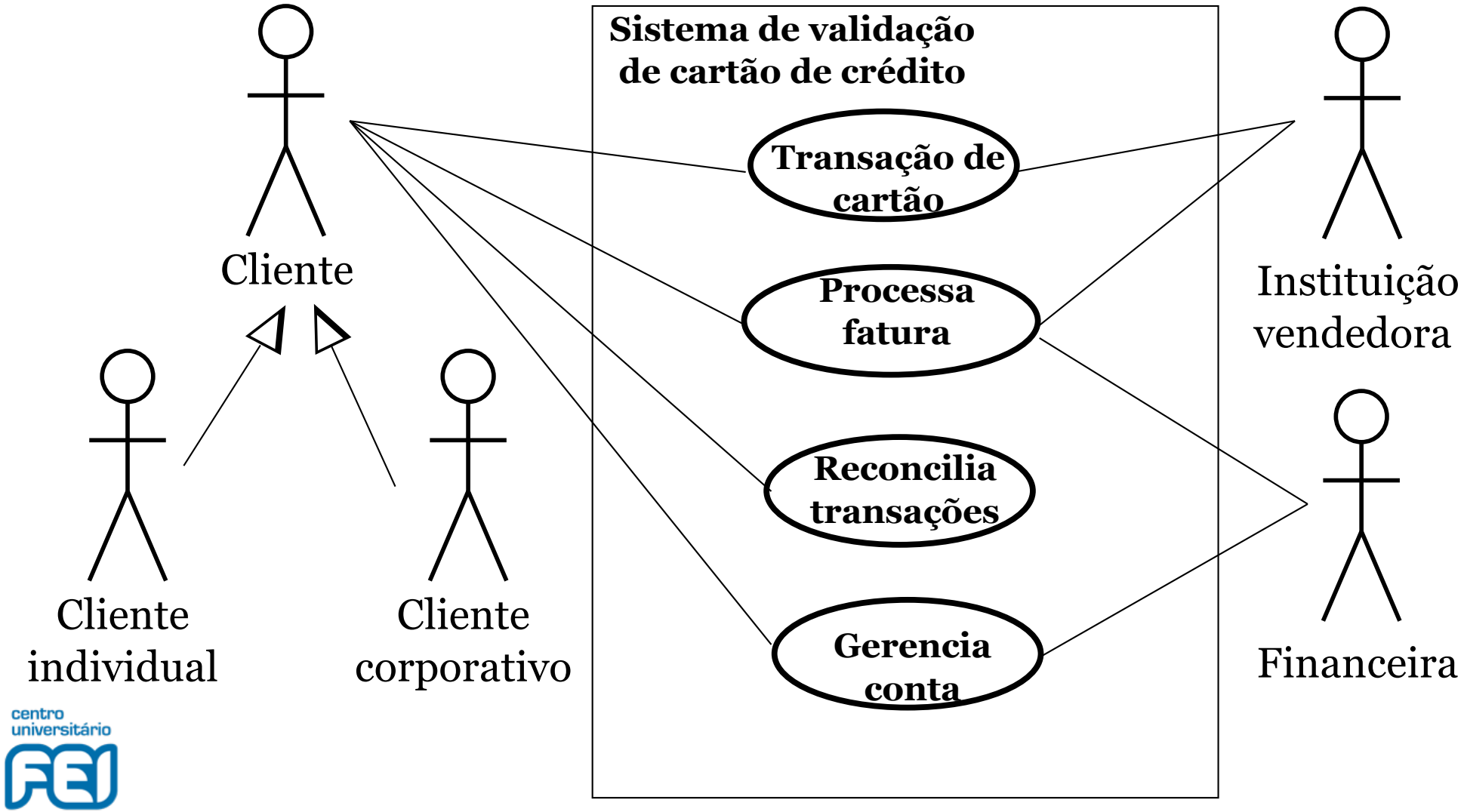
  - Se Cliente entra com senha inválida, o *use case* reinicia.

Diagramas de *Use Cases*

- Com o intuito de demarcar os limites da funcionalidade do sistema, *use cases* são organizados dentro de um diagrama;

- Quando do uso de diagramas de *use cases*, os atores são modelados através de relacionamentos de generalização/especialização;

Exemplo de Diagrama



Exemplo de Caso de uso

- Realizar um saque no caixa eletrônico

Identificação	UC_01
Função	Retirar Dinheiro do caixa eletrônico
Atores	Cliente, Caixa eletrônico
Prioridade	Essencial
Pré-condição	Cliente precisa ter em mãos o cartão do banco
Pós-condição	Dinheiro sacado com sucesso
Fluxo Principal	•Cliente insere cartão no dispositivo •Cliente digita a senha •Máquina autoriza login [FS001] •Cliente digita o montante •Máquina checa o saldo [FS002] •Máquina debita o dinheiro sacado do saldo inicial •Máquina dispõe cédulas para cliente •Máquina mostra na tela no novo saldo •Máquina ejeta cartão •Cliente retira cartão

Fluxo Secundário [FS001]	•Senha digitada é inválida •Máquina ejeta cartão •Cliente retira cartão

Fluxo Secundário [FS002]	•Saldo é menor que o montante requerido •Máquina mostra na tela o saldo •Máquina ejeta o cartão •Cliente retira o cartão

Dica Final: <<*Include*>>

- Use inclusão quando o mesmo comportamento se repete em mais de um caso de uso.
  - Esse comportamento comum deve ser definido em um novo caso de uso, o chamado caso de uso incluído.
  - Note que esse comportamento comum está contido em todos os cenários dos casos de uso.
  - A ligação ocorre com seta saindo dos casos de uso "inclusor" para o "incluído".
  - O "incluído" é esse que você criou devido o comportamento se repetir.

centro universitário FEI

Dica Final: <<extend>>

- Use extensão quando um comportamento opcional de um caso de uso tiver de ser descrito.
  - Note que alguns cenários de caso de uso estendido podem não utilizar esse comportamento opcional.
  - O extensor (aquele caso de uso opcional) faz referência (ligação da seta) ao estendido
  - sendo que o estendido não sabe que o extensor existe.