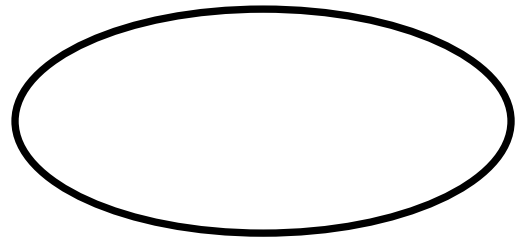


# Diagramas de Casos de Uso

# Objetivos

- Introduzir conceitos de *use case*, ator e fluxo de eventos
- Apresentar sub-fluxos de eventos
- Discutir sobre identificação, evolução e organização de *use cases*
- Apresentar notação UML para reusar atores e *use cases*

# *Use Case*

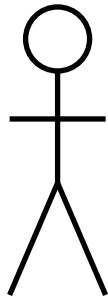


Função

- Seqüência de ações, executada pelo sistema, que gera um resultado
  - De valor observável
  - E para ator particular

Procedimento computacional/algorítmico atômico

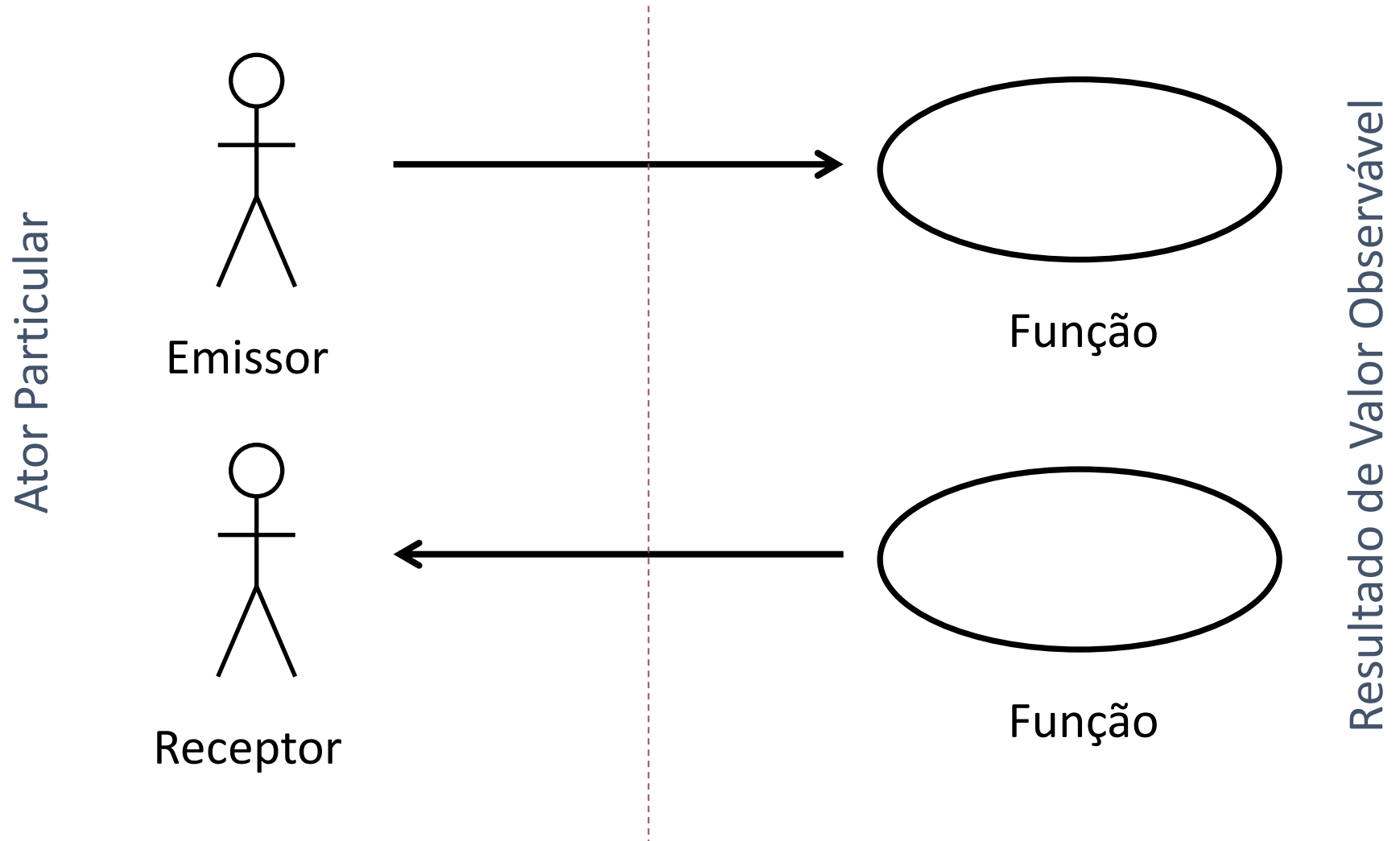
# *Use Case e Ator*



Emissor/Receptor

- Alguém ou alguma coisa (**fora do sistema**) que interage com o sistema

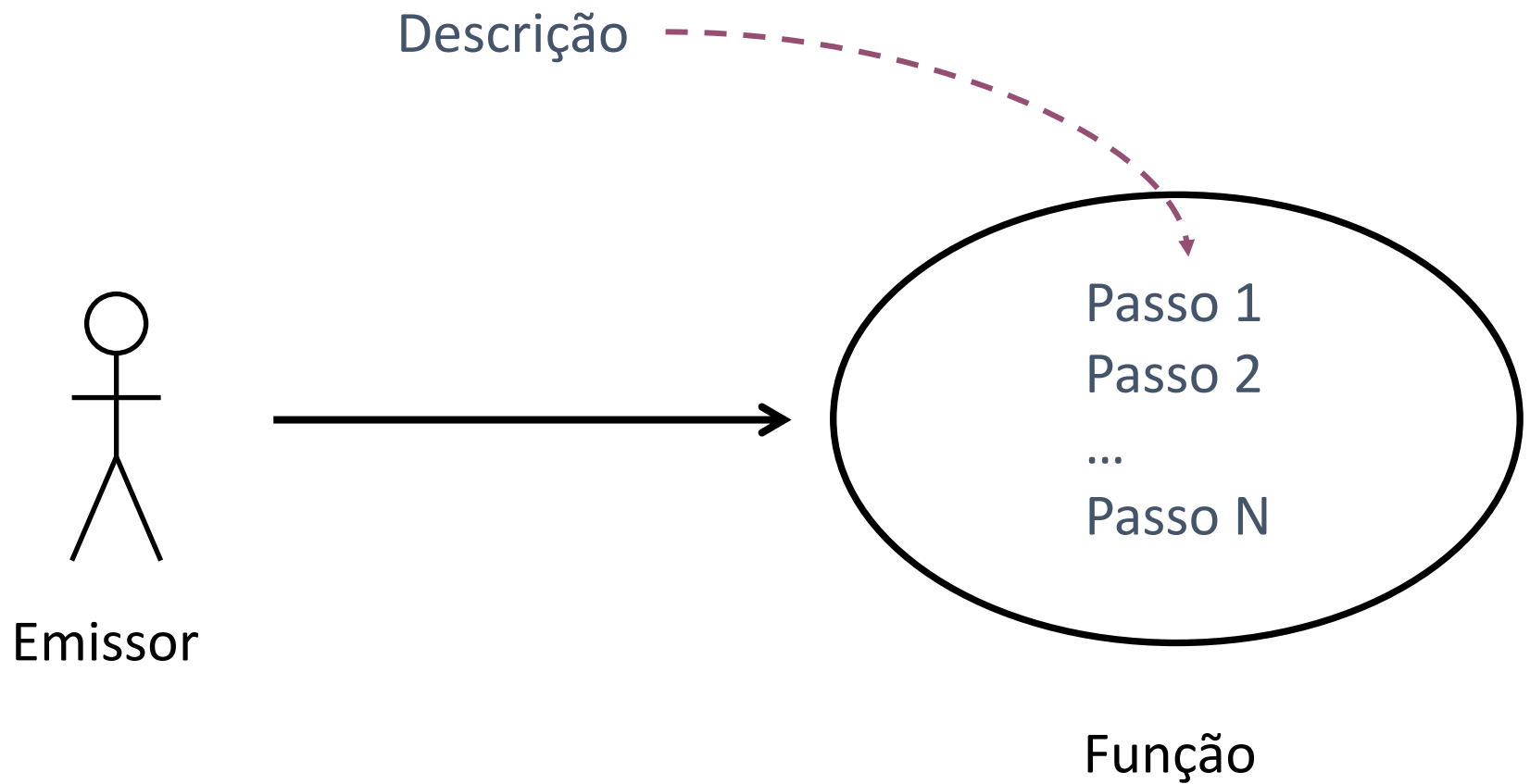
## *Use Case e Ator*



# *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

# Use Case e Ator

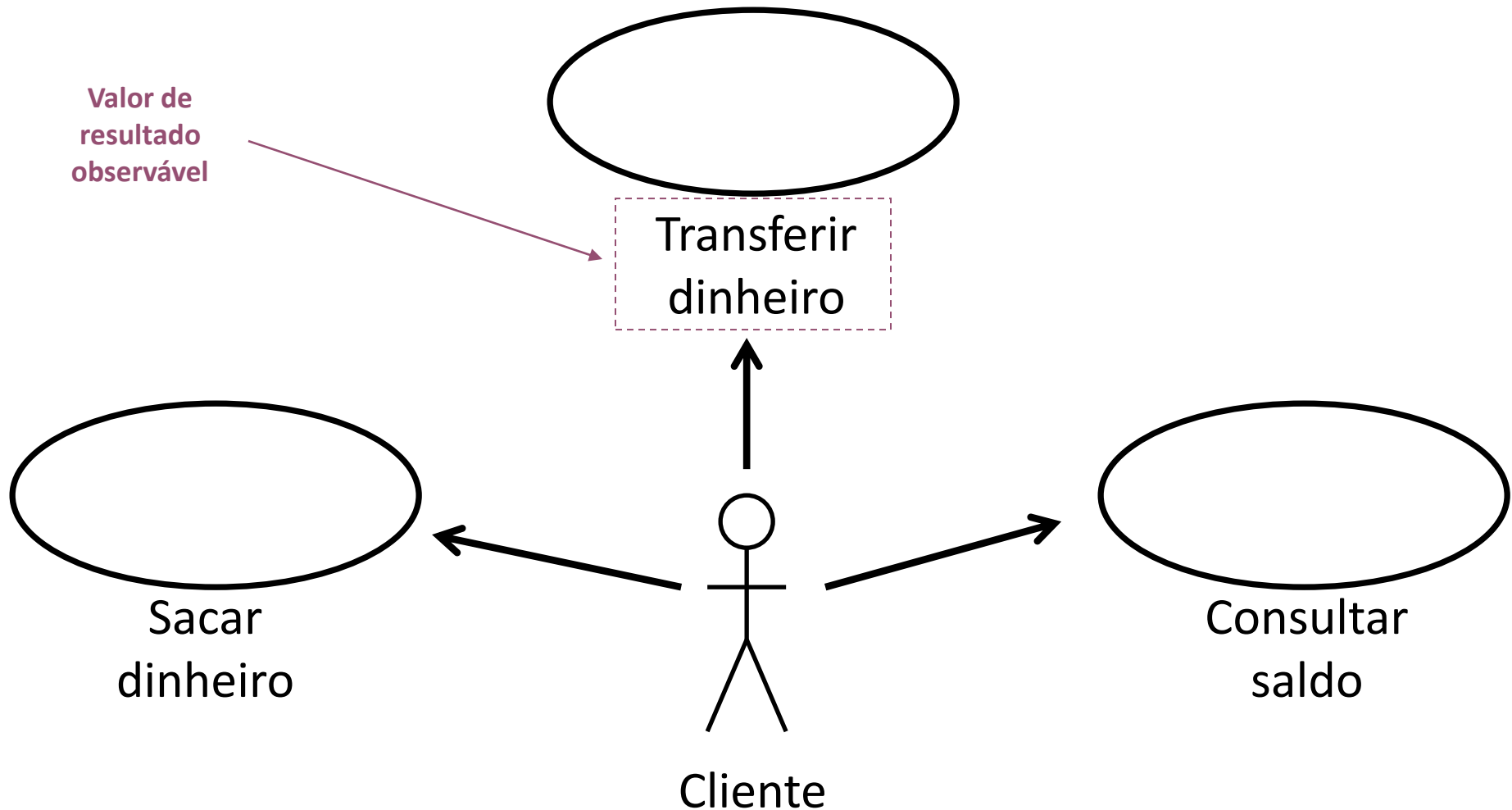


# Exemplo de *Use Case* e Ator

- Cliente de banco pode usar um caixa automático 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, difícil decidir entre um ou vários use cases
- Por exemplo, seriam use cases
  - Inserir cartão em um Caixa Automático?
  - Entrar com a senha?
  - Receber o cartão de volta?

# Identificando *Use Cases*

- Representar valor observável para ator
- Pode-se determinar
  - De interações (seqüência de ações) com o sistema que resultam valores para atores
  - Satisfaz um objetivo particular de um ator que o sistema deve prover

# Identificando *Use Cases*

- Facilitar gerenciamento durante ciclo de desenvolvimento
  - A razão para agrupar funcionalidades e chamá-las de *use cases*

# Exercício

- Tenho um sistema que é acionado 2 vezes por dia (às 10:20hs e 17:20hs), enviando-me um SMS. Também tenho como obter resultado semelhante acessando tal funcionalidade a partir de meu celular (web browser). Crie os casos de uso correspondentes.

# Evolução de *Use Cases*

- Inicialmente *use cases* são simples
  - Apenas esboço sobre funcionamento é suficiente
- Mas com a sedimentação da modelagem
  - Descrição mais detalhada do fluxo de eventos faz-se necessária
- Fluxo de eventos deve ser refinado
  - Todos os *stakeholders* envolvidos devem estar de acordo com a descrição

# Organizando *Use Cases*

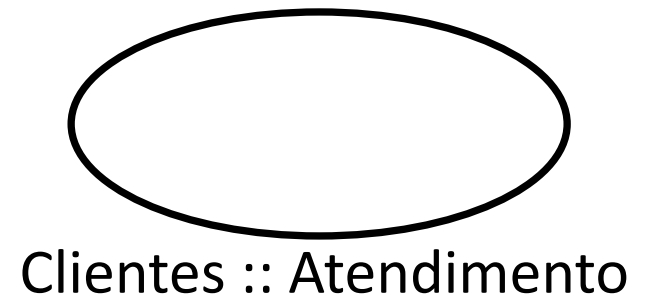
- Sistema pequeno não demanda estruturação
  - Exemplo, seis *use cases*, com dois/três atores
- Já sistemas maiores requerem princípios de estruturação e organização
  - Caso contrário, planejamento, atribuição de prioridades, etc., podem se tornar difíceis

# Pacote de *Use Case*

- Primeiro esforço de estruturação
- Agrupam-se use cases relacionados em único *container*



# Pacote de *Use Cases*



# Reuso em *Use Cases*

- Comportamento comum a mais de dois *use cases* (ou forma parte independente)
  - Pode-se modelar como *use case* para ser reusado
- Há três possibilidades
  - Inclusão
  - Extensão
  - Generalização/Especialização

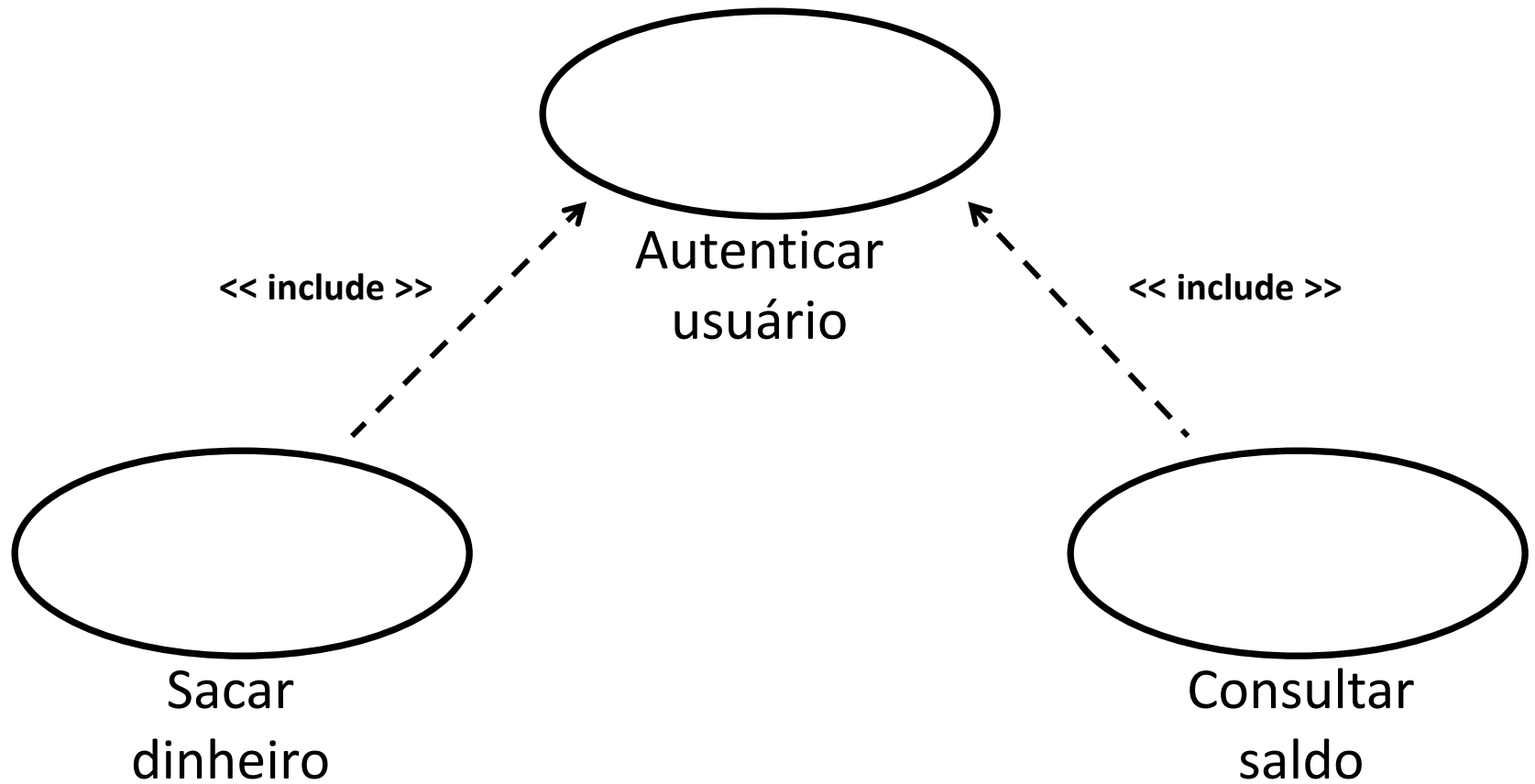
# Inclusão

- Vários use cases possuem texto de fluxo de eventos
  - Comum/idêntico
  - Sempre usado
- Equivalente a faturação feita em programação através de sub-programas
  - #include da linguagem C

# Inclusão

- Como exemplo, tanto “Sacar dinheiro” quanto “Consultar saldo” necessitam da senha
  - Pode-se criar novo use case “Autenticar usuário” e incluí-lo
- Mas atenção
  - NÃO SE DEVE CRIAR USE CASES MÍNIMOS
  - Deve haver ganho no reuso

# Inclusão



# Inclusão

- Descrição de **Consultar saldo**
  - Fluxo de Eventos Principal:
    - **include (Autenticar usuário)**. Sistema pede a Cliente que selecione tipo de conta (corrente, etc). ...

# Extensão

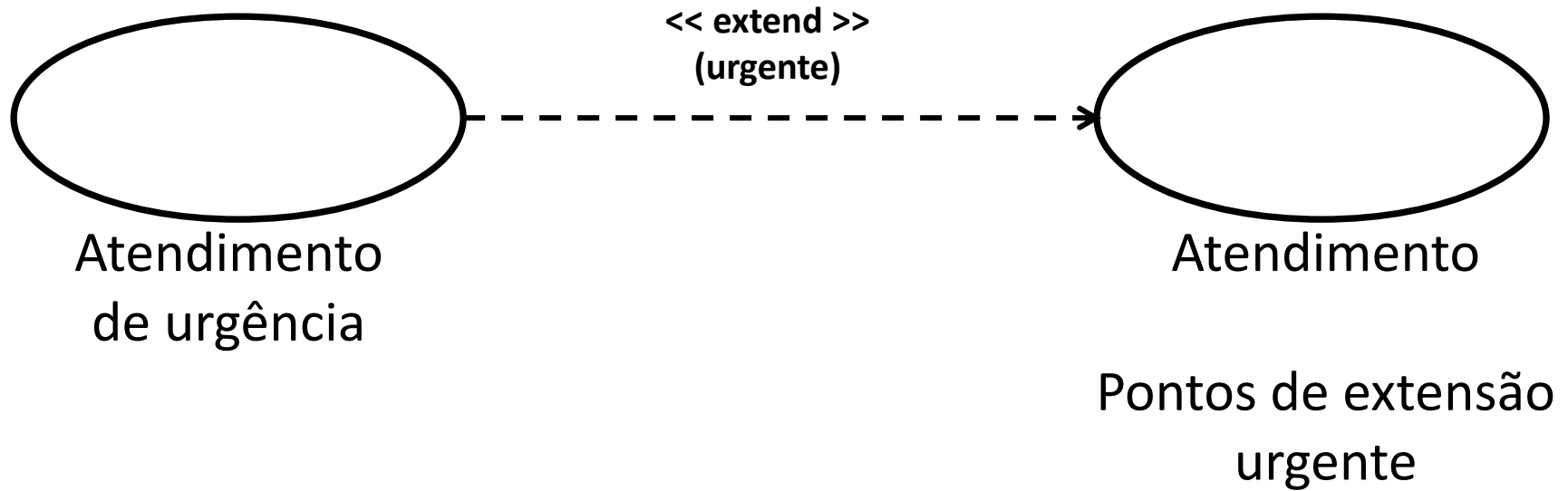
- Use case pode ser estendido por outro
  - Extensão de funcionalidade/Caso excepcional
- Extensão ocorre em pontos específicos
  - Pontos de extensão

# Extensão

- Há também inclusão de texto (fluxo de eventos)
  - Porém sob condições particulares
- Pode ser usada para
  - Simplificar fluxos de eventos complexos
  - Representar comportamentos opcionais
  - Lidar com exceções



# Extensão



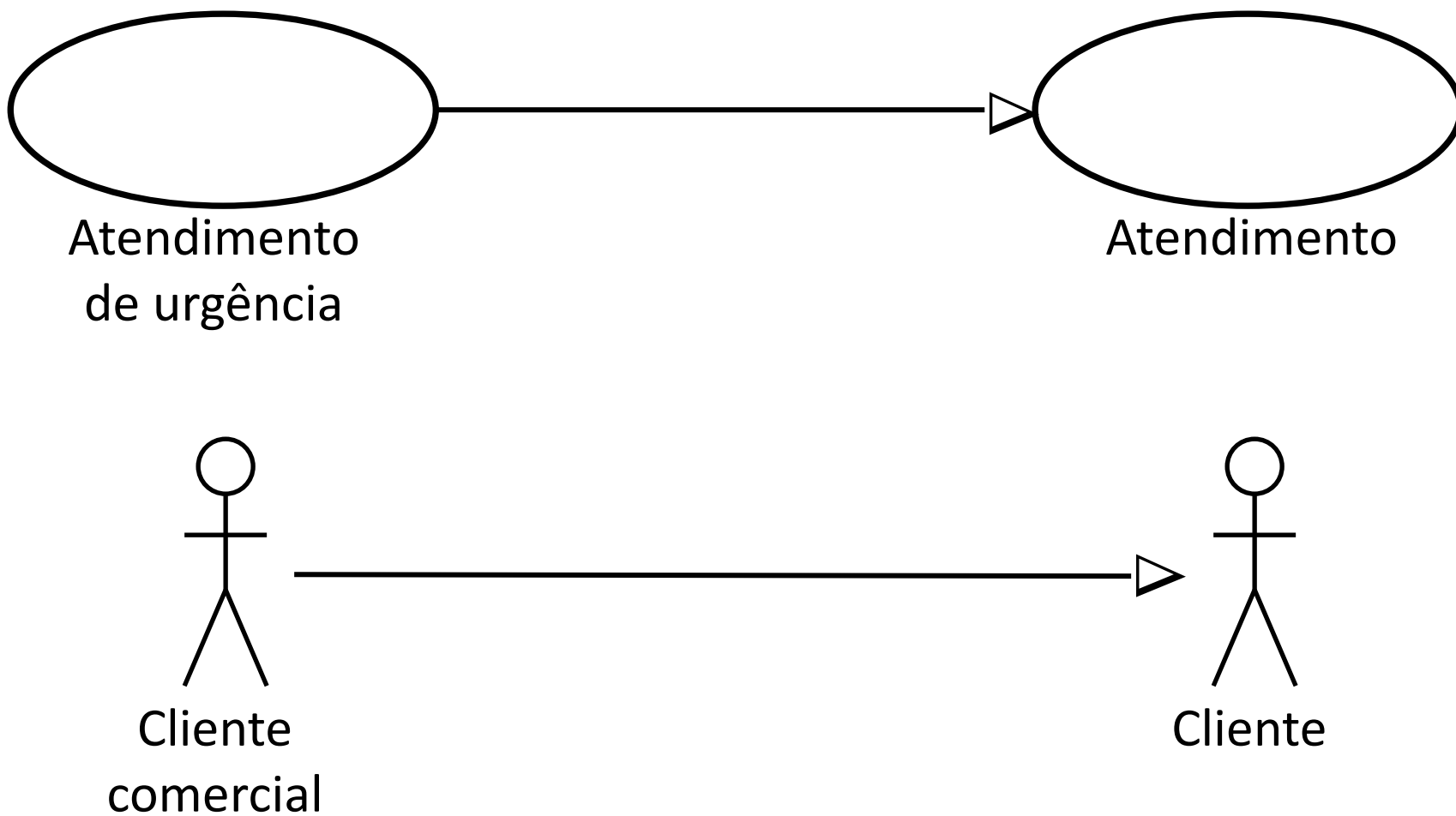
# Extensão

- Descrição de **Atendimento**
  - Fluxo de Eventos Principal:
    - Colete os itens do pedido. (**urgente**). Submeta pedido para processamento.

# Especialização

- Use *case* pode especializar outro
  - Adição/refinamento do fluxo de eventos original
- Especialização permite modelar comportamento de estruturas de aplicação em comum

# Especialização



# Fluxo de Eventos

- Parte mais importante de um use case
  - Atividade de requisitos
- Define a seqüência de ações entre o ator e o sistema

# Fluxo de Eventos

- Geralmente em linguagem natural
  - Uso preciso de termos definidos no glossário de acordo com o domínio do problema
- Também expresso formalmente
  - Pré e pós-condições (ou pseudo-código)

# 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”

# Exemplo de Fluxo de Eventos

- Um esboço inicial sobre Sacar dinheiro seria
  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.



# Exemplo de Fluxo de Eventos

- Um esboço inicial sobre Sacar dinheiro seria
  7. Sistema pede remoção do cartão. Cliente remove.
  8. Sistema entrega quantia solicitada.

# Fluxo de Eventos

- Na descrição do que o sistema faz através de fluxos de eventos completos
  - Surgem caminhos alternativos
  - Casos diferentes a considerar
  - Efeitos/valores diferentes a produzir
- Eventualmente descreve todos esses caminhos possíveis

# Sub-fluxos de Eventos

- Fluxo de eventos visto como
  - Vários sub-fluxos de eventos
- Sub-fluxos são descritos como
  - Principal
  - Alternativos/excepcionais
- Abordagem visa reuso de fluxos de eventos (sub-fluxos)

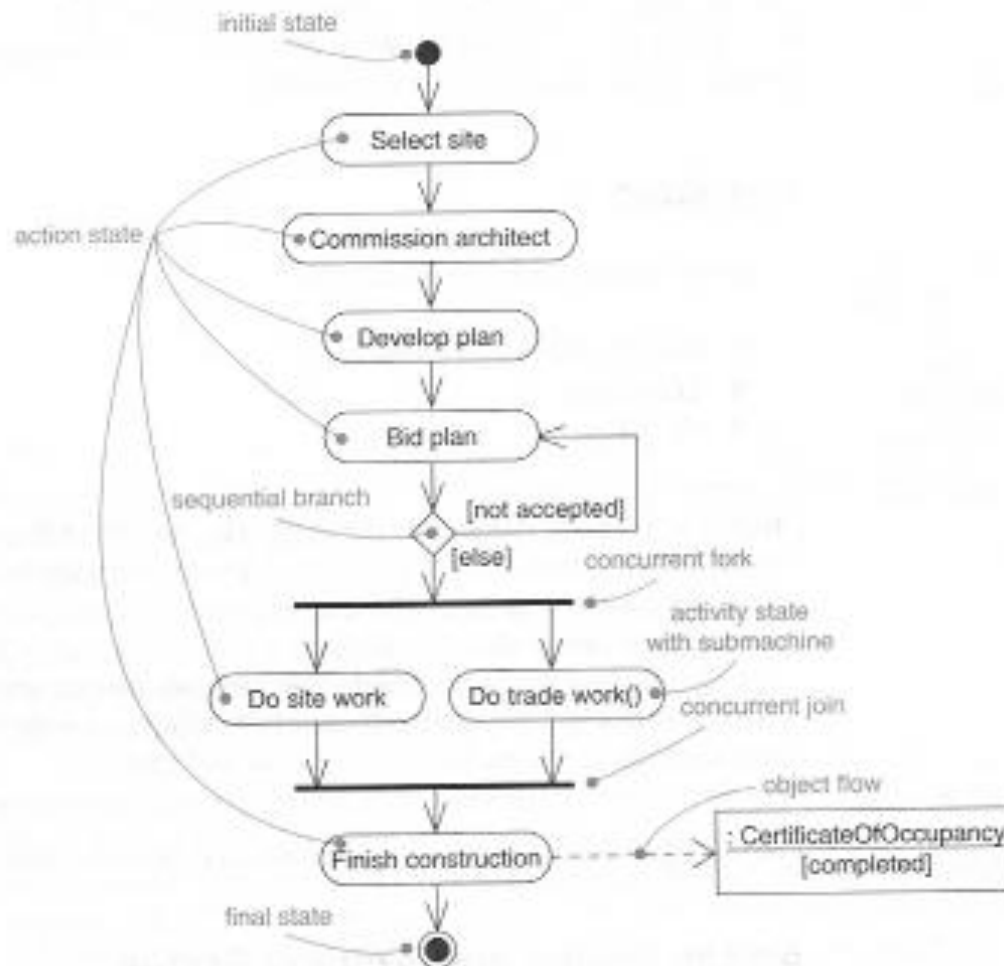
# Exemplo de Sub-fluxos

- 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.

# Diagrama de Atividades

- Descreve fluxo de tarefas
  - Aspectos dinâmicos de um sistema
  - Podem também ser usados para criar sistemas executáveis
    - Depende do nível de detalhamento e grau de execução dos elementos usados
- Alternativa para modelar fluxos de eventos de casos de uso

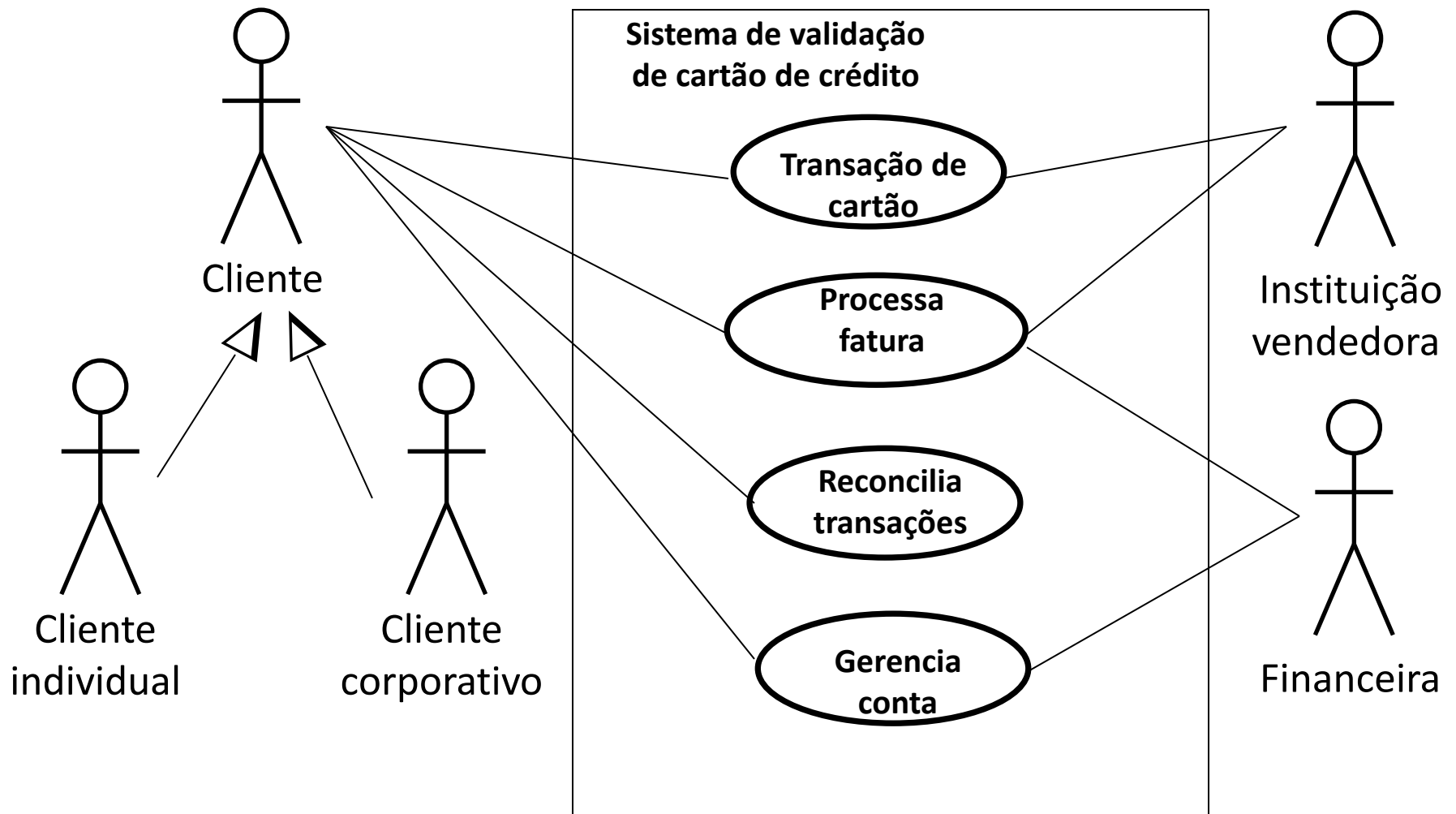
# Elementos de um Diag. Ativ.



# Diagramas de *Use Cases*

- Caracterizar limites da funcionalidade do sistema
  - *Use cases* são organizados dentro de um diagrama
- Em diagramas de *use cases*
  - Atores são relacionados por generalização/especialização

# Exemplo de Diagrama





# Bibliografia

- Sommerville, I. Software Engineering
- Kruchten, P. The Rational Unified Process: An Introduction. 2nd Ed
- Booch, G. et al. The Unified Modeling Language User Guide.