

# Engenharia de Software

**Professor:**

Zady Castaneda Salazar



INSTITUTO FEDERAL DE  
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA  
SÃO PAULO  
Campus Campinas

# Aula 15 - UML



INSTITUTO FEDERAL DE  
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA  
SÃO PAULO  
Campus Campinas

# UML Unified Modeling Language

---

Na área de Engenharia de Software, a Linguagem de Modelagem Unificada (do inglês, **UML - Unified Modeling Language**) é uma **linguagem de modelagem** que permite representar um sistema de forma padronizada.

# UML Unified Modeling Language

---

## Vantagem:

Permite que desenvolvedores visualizem os produtos de seus trabalhos em diagramas padronizados.

É única para as quatro atividades: análise, “design”, implementação e teste.

# UML Unified Modeling Language

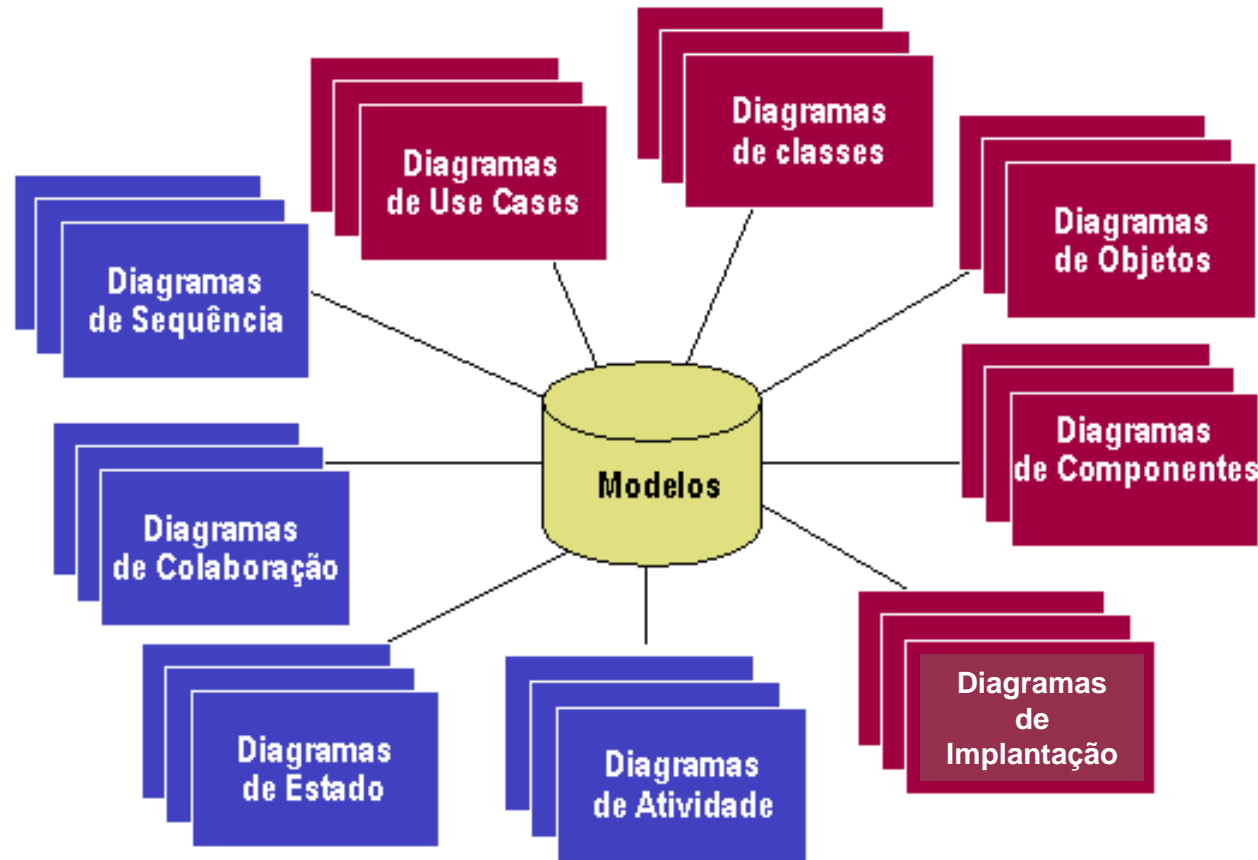
---

## Tipos dos Diagramas UML

- **Diagrama estáticos**
  - **Diagrama de casos de uso**
  - Diagrama de classes
  - Diagrama de objetos
  - Diagrama de componentes
  - Diagrama de implantação
- **Diagrama dinâmicos**
  - Diagrama de sequência
  - Diagrama de colaborações
  - Diagrama de estados
  - Diagrama de atividades

# UML Unified Modeling Language

O modelo do sistema é representado pelos dois conjuntos de diagrama, estático e dinâmico(ver figura abaixo).



*Fig1: Vermelho: estático (ou estrutural); Azul: dinâmico (ou comportamental).*

# UML Unified Modeling Language

---

Um **diagrama** provê uma parcial representação do sistema.

Ajuda a compreender a arquitetura do sistema em desenvolvimento.

# Diagramas Estáticos: Casos de usos

---

O diagrama de casos de uso é o diagrama mais geral e informal da UML, utilizado normalmente nas fases de levantamento e análise de requisitos do sistema, embora venha a ser consultado durante todo o processo de modelagem e possa servir de base para outros diagramas.

Apresenta uma linguagem simples e de fácil compreensão para que os usuários **possam ter uma ideia geral de como o sistema irá se comportar.**



# Diagramas Estáticos: Casos de usos

---

O diagrama de Casos de Uso tem o **objetivo de auxiliar a comunicação entre os analistas e o cliente.**

Um diagrama de Caso de Uso **descreve um cenário que mostra as funcionalidades do sistema do ponto de vista do usuário.**

O cliente deve ver no diagrama de Casos de Uso as principais funcionalidades de seu sistema.

# Diagramas Estáticos: Casos de usos

---

## Notação

O diagrama de Caso de Uso é representado por:

- atores;
- casos de uso;
- relacionamentos entre estes elementos.

Estes relacionamentos podem ser:

- associações
- generalizações
- *extends* e *includes*

# Diagramas Estáticos: Casos de usos

## Exemplo

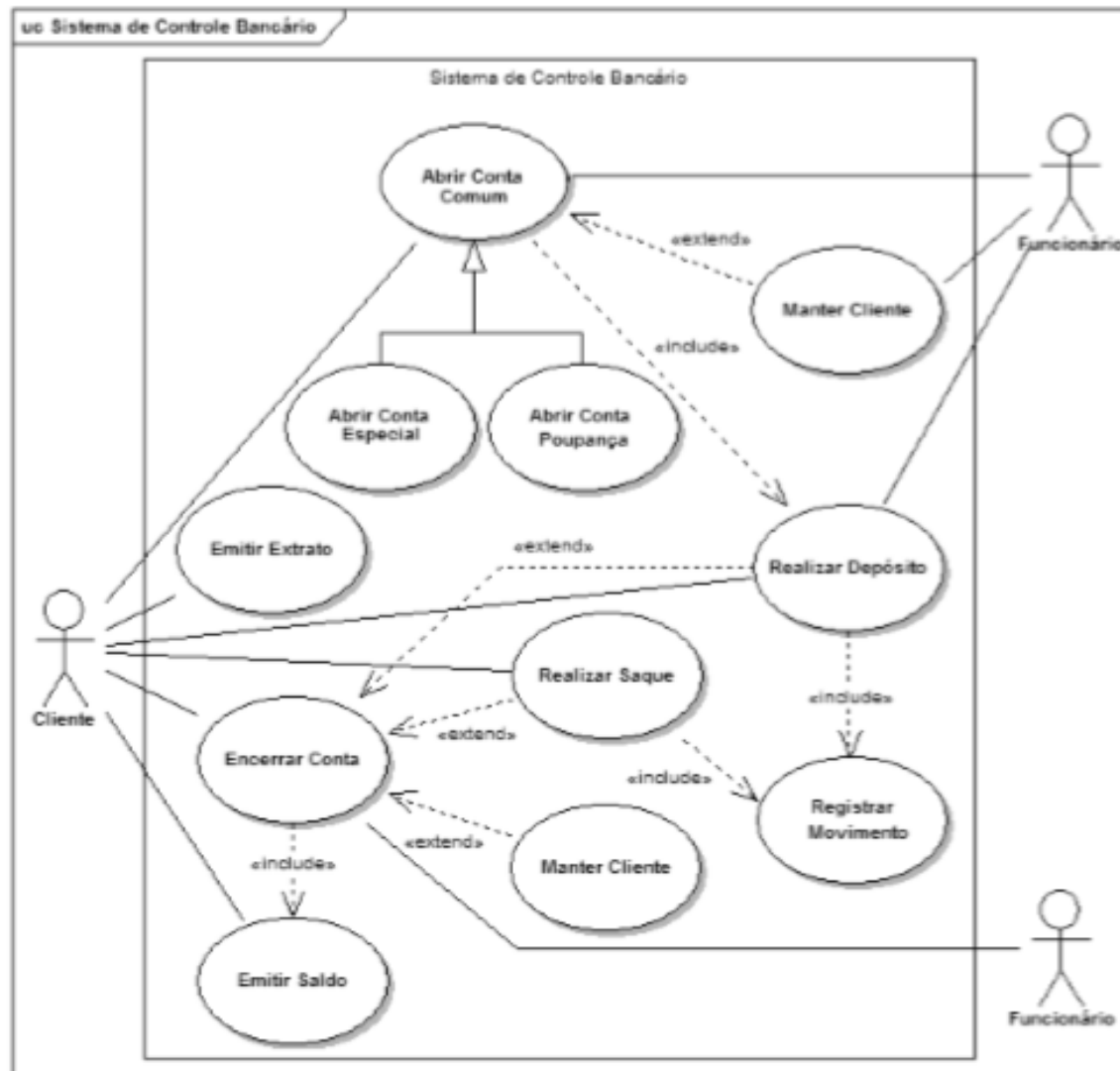


Figura 1.1 – Exemplo de Diagrama de Casos de Uso.

# Diagramas Estáticos: Diagrama de Classes

---

O diagrama de classes é provavelmente o mais utilizado e é um dos mais importantes da UML. Serve de apoio para a maioria dos demais diagramas.

Como o próprio nome diz, **define a estrutura das classes utilizadas pelo sistema**, determinando os atributos e métodos que cada classe tem, além de estabelecer como as classes se relacionam e trocam informações entre si.

# Diagramas Estáticos: Diagrama de Classes

---

## Objetivo

Descrever os vários tipos de objetos no sistema e o relacionamento entre eles.

Um diagrama de classes contém:

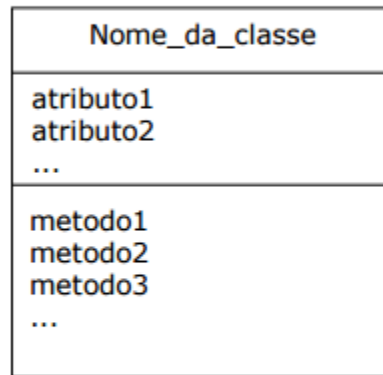
- Classes
- Relacionamentos
- Notações

# Diagramas Estáticos: Diagrama de Classes

---

## CLASSE

Graficamente, as classes são representadas por retângulos incluindo nome, atributos e métodos.



Devem receber nomes de acordo com o vocabulário do domínio do problema.

É comum adotar um padrão para nomeá-las

**Exemplo:** todos os nomes de classes serão substantivos singulares com a primeira letra maiúscula

# Diagramas Estáticos: Diagrama de Classes

---

## Classe

### Atributos

- Representam o conjunto de características (estado) dos objetos daquela classe
- Visibilidade:
  - + público: visível em qualquer classe de qualquer pacote
  - # protegido: visível para classes do mesmo pacote
  - privado: visível somente para classe

### Métodos

- Representam o conjunto de operações (comportamento) que a classe fornece
- Visibilidade:
  - + público: visível em qualquer classe de qualquer pacote
  - # protegido: visível para classes do mesmo pacote
  - privado: visível somente para classe

# Diagramas Estáticos: Diagrama de Classes

---

## RELACIONAMENTOS

Os relacionamentos possuem:

- **Nome:** descrição dada ao relacionamento (faz, tem, possui,...)
- **Sentido de leitura**
- **Navegabilidade:** indicada por uma seta no fim do relacionamento
- **Multiplicidade:** 0..1, 0..\*, 1, 1..\*, 2, 3..7
- **Tipo:** associação (agregação, composição), generalização e dependência
- **Papéis:** desempenhados por classes em um relacionamento



# Diagramas Estáticos: Diagrama de Classes

---

## RELACIONAMENTOS

- Multiplicidade (utilizado em todas as perspectivas de forma uniforme)

Notações possíveis


<b><i>Tipos</i></b>	<b><i>Significa</i></b>
0..1	Zero ou uma instância. A notação n..m indica n para m instâncias.
0..* ou *	Não existe limite para o número de instâncias.
1	Exatamente uma instância.
1..*	Ao menos uma instância.

# Diagramas Estáticos: Diagrama de Classes

---

## RELACIONAMENTOS

### TIPOS

- 
- Associação
    - Agregação
    - Composição
  - Generalização
  - Dependência

# Diagramas Estáticos: Diagrama de Classes

---

## RELACIONAMENTOS: Associação

- Uma **associação** é um relacionamento estrutural que indica que os objetos de uma classe estão vinculados a objetos de outra classe.
- Uma associação é representada por uma linha sólida conectando duas classes.



# Diagramas Estáticos: Diagrama de Classes

## RELACIONAMENTOS: Associação

Indicadores de multiplicidade:

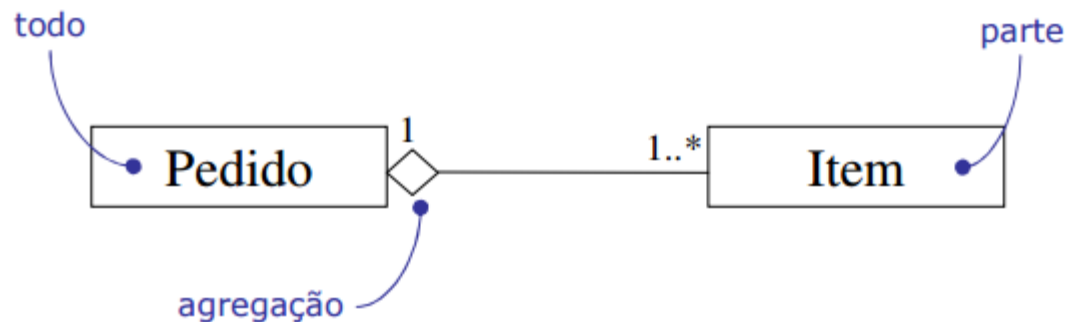
- 1 Exatamente um
- 1..\* Um ou mais
- 0..\* Zero ou mais (muitos)
- \* mais (muitos)
- 0..1 Zero ou um
- m..n Faixa de valores (por exemplo: 4..7)



# Diagramas Estáticos: Diagrama de Classes

## RELACIONAMENTOS: Agregação

É um tipo especial de associação  
Utilizada para indicar “todo-parte”



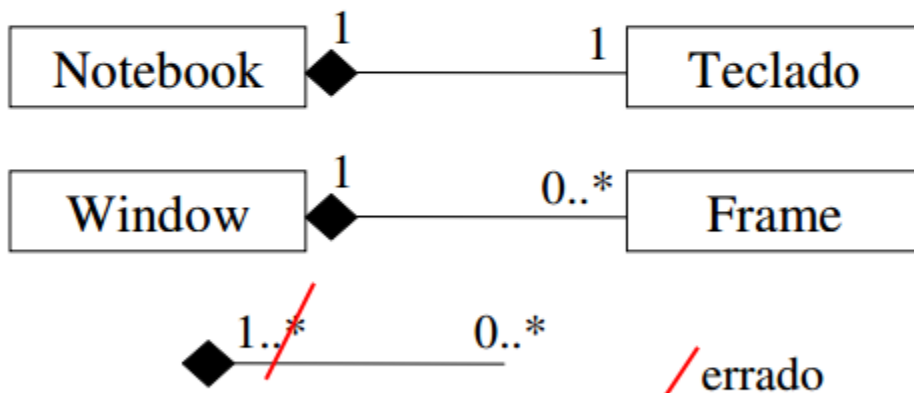
- um objeto “parte” pode fazer parte de vários objetos “todo”

# Diagramas Estáticos: Diagrama de Classes

## RELACIONAMENTOS: Composição

É uma variante semanticamente mais “forte” da agregação

Os objetos “parte” só podem pertencer a um único objeto “todo” e têm o seu tempo de vida coincidente com o dele

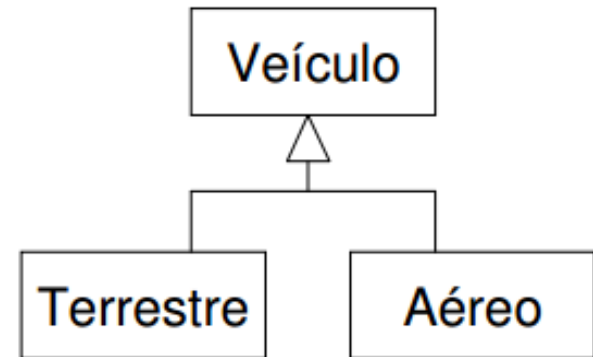
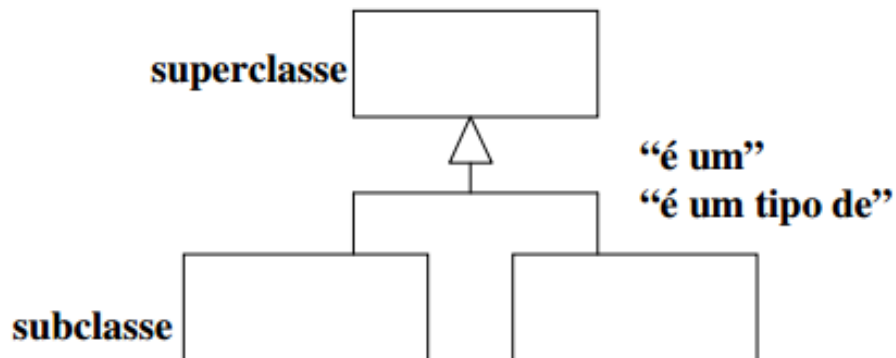


- Quando o “todo” *morre* todas as suas “partes” também *morrem*

# Diagramas Estáticos: Diagrama de Classes

## RELACIONAMENTOS: Generalização

- É um relacionamento entre itens gerais (superclasses) e itens mais específicos (subclasses)



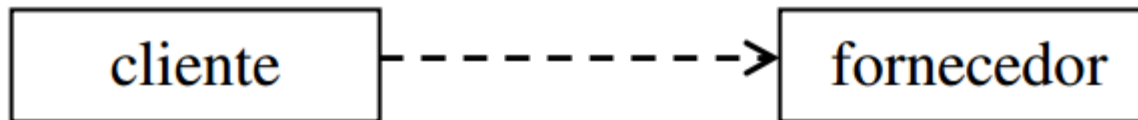
# Diagramas Estáticos: Diagrama de Classes

---

## RELACIONAMENTOS: Dependência

- Representa que a alteração de um objeto (o objeto independente) pode afetar outro objeto (o objeto dependente)

Ex:



Obs:

- A classe cliente depende de algum serviço da classe fornecedor
- A mudança de estado do fornecedor afeta o objeto cliente
- A classe cliente não declara nos seus atributos um objeto do tipo fornecedor
- Fornecedor é recebido por parâmetro de método



# Diagramas Estáticos: Diagrama de Classes

---

## Perspectivas

Um diagrama de classes pode **oferecer três perspectivas**, cada uma para um tipo de usuário diferente.

São elas:

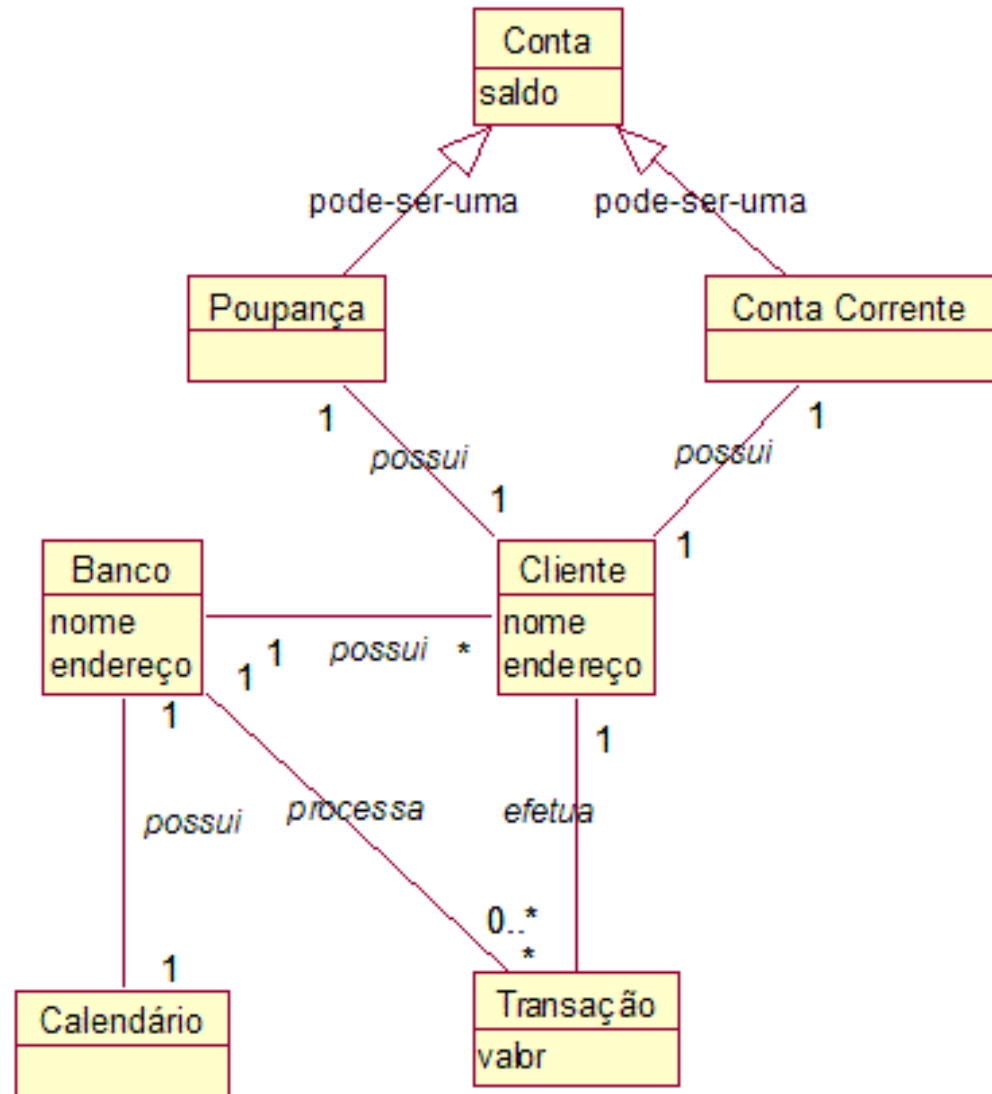
- ☐ Conceitos e entidades
- ☐ Classes
- ☐ Classes de software

# Diagramas Estáticos: Diagrama de Classes

## Perspectivas

### ❑ Conceitos e entidades

- Representa os conceitos do domínio em estudo.
- Perspectiva destinada ao cliente

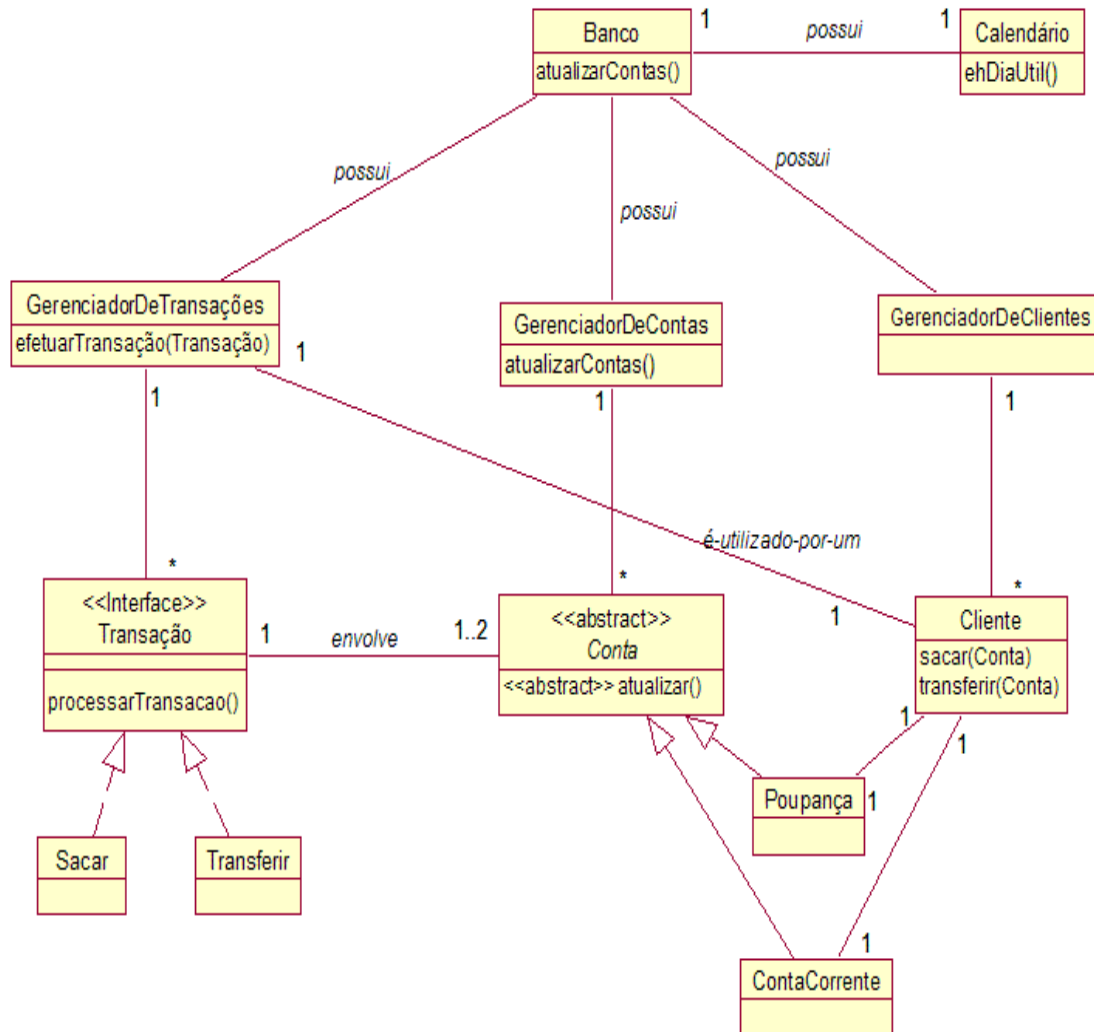


# Diagramas Estáticos: Diagrama de Classes

## Perspectivas

### □ Classes

- Tem foco nas principais interfaces da arquitetura, nos principais métodos.
- Perspectiva destinada as pessoas que não precisam saber detalhes de desenvolvimento, tais como gerentes de projeto.

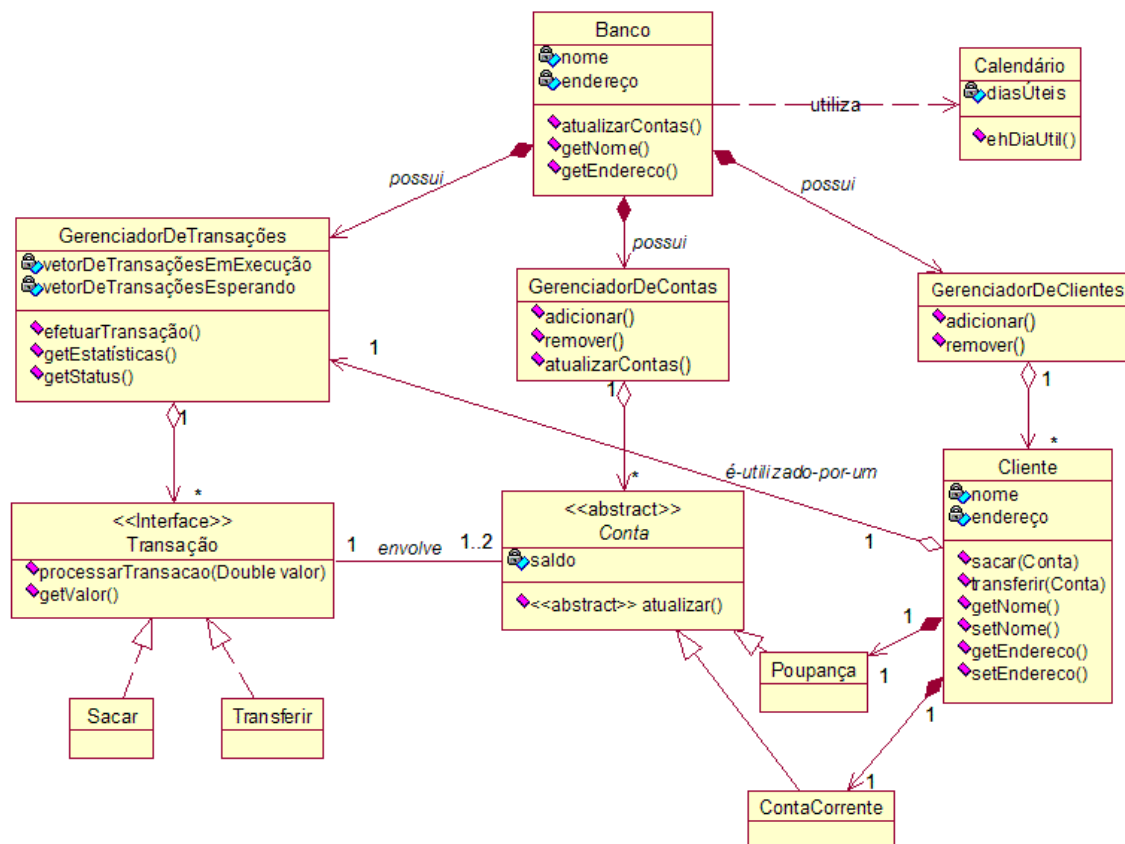


# Diagramas Estáticos: Diagrama de Classes

## Perspectivas

### ☐ Classes de software

- Aborda vários detalhes de implementação, tais como navegabilidade, **tipo** dos atributos, etc.
- Perspectiva destinada ao time de desenvolvimento.



# Diagramas Estáticos: Diagrama de objeto

---

O diagrama de objetos está **amplamente associado ao diagrama de classes**. Na verdade, o diagrama de objetos é praticamente um complemento do diagrama de classes e bastante dependente deste.

O diagrama **fornece uma visão dos valores armazenados pelos objetos de um diagrama de classes em um determinado momento da execução de um processo do software**.

# Diagramas Estáticos: Diagrama de objeto, Exemplo

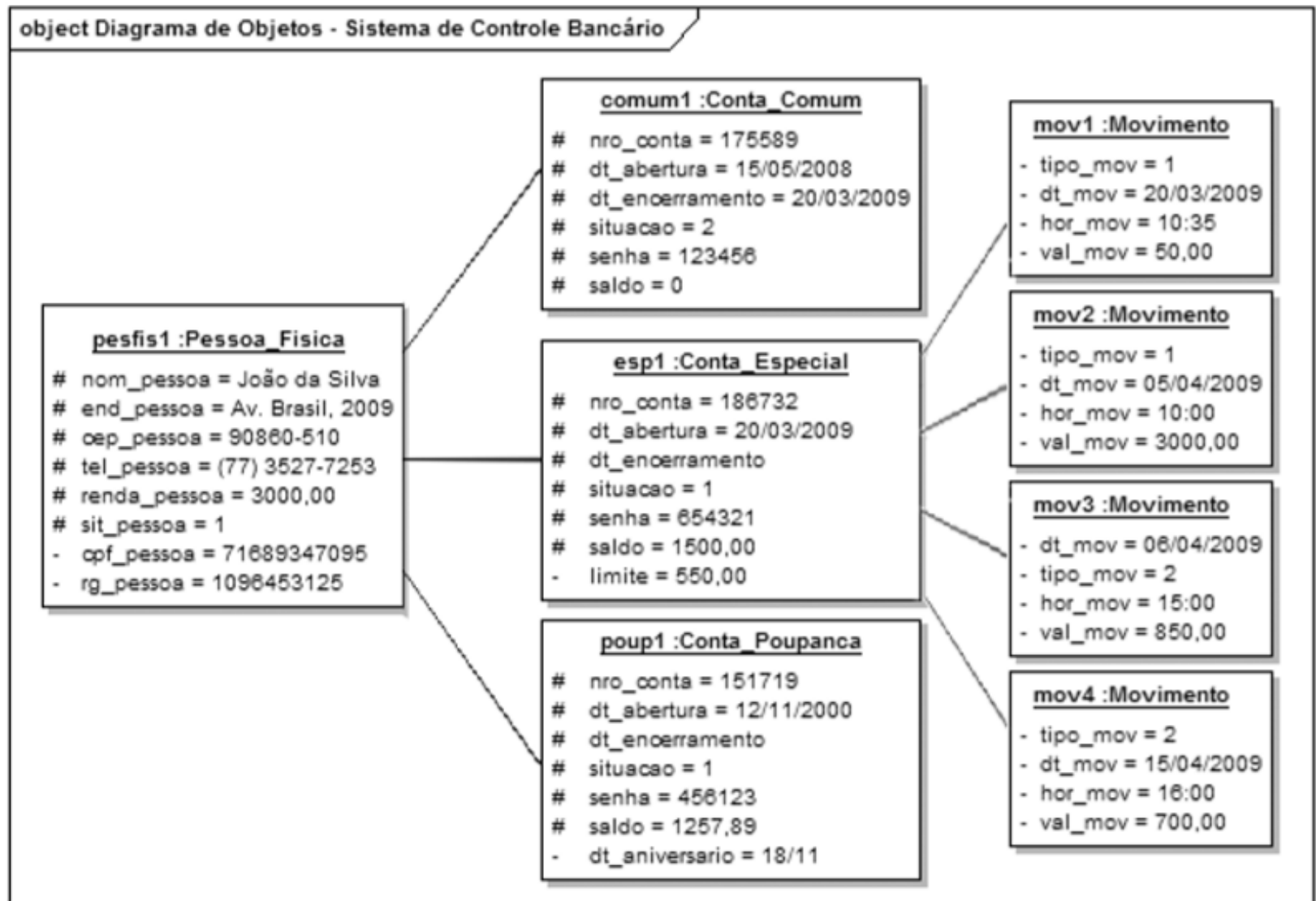


Figura 1.3 – Exemplo de Diagrama de Objetos.

# Diagramas Estáticos: Diagrama de Componentes

---

Um componente é uma peça básica na implementação de um sistema; consiste, na prática, num conjunto de artefatos físicos em formato digital, por exemplo, arquivos de código (fonte, binário ou executáveis) ou arquivos de documentos relativos ao negócio.

# Diagramas Estáticos: Diagrama de Componentes

---

## Tipos de componentes:

- Componentes de instalação: constituem a base dos sistemas executáveis (e.g., DLL, executáveis, controles Active-X, classes Java).
- Componentes de trabalho: a partir dos quais são criados os componentes de instalação (e.g., arquivos com código fonte, arquivos de dados, documentos).
- Componentes de execução: criados como resultado da execução de um sistema (e.g., processos, threads, agentes de software).



# Diagramas Estáticos: Diagrama de Componentes

---

O diagrama de componentes está amplamente **associado à linguagem de programação que será utilizada para desenvolver o sistema modelado.**

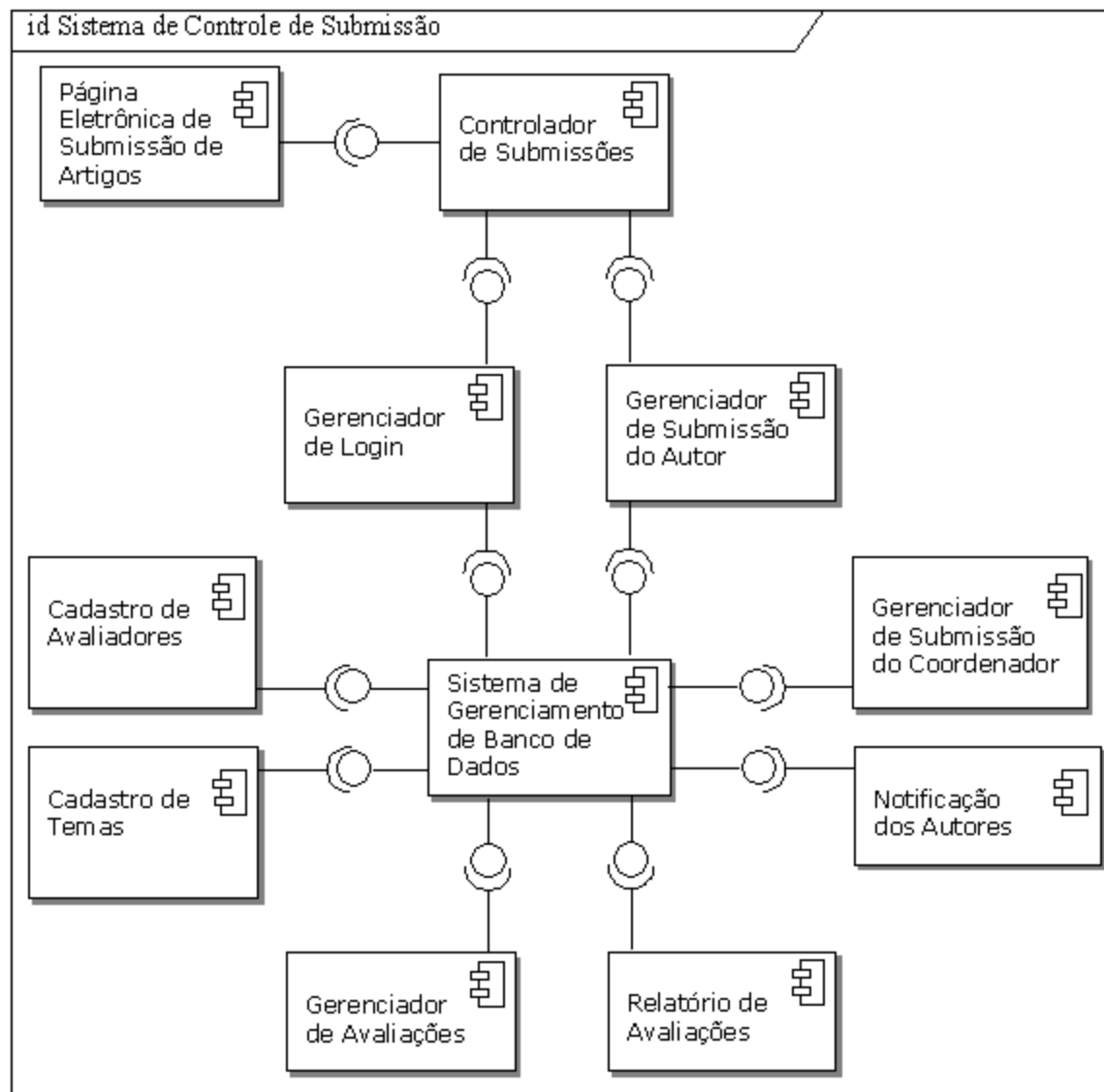
Esse diagrama **representa os componentes do sistema quando o mesmo for ser implementado** em termos de módulos de código-fonte, bibliotecas, formulários, arquivos de ajuda, módulos executáveis etc. e determina como tais componentes estarão estruturados e irão interagir para que o sistema funcione de maneira adequada

# Diagramas Estáticos: Diagrama de Componentes

Un componente usa una interfaz



## UML 2



# Diagramas Estáticos: Diagramas de implantação

---

O diagrama de implantação **determina as necessidades de hardware do sistema**, as características físicas como servidores, estações, topologias e protocolos de comunicação, ou seja, todo o aparato físico sobre o qual o sistema deverá ser executado.

# Diagramas Estáticos: Diagramas de implantação

## Exemplo

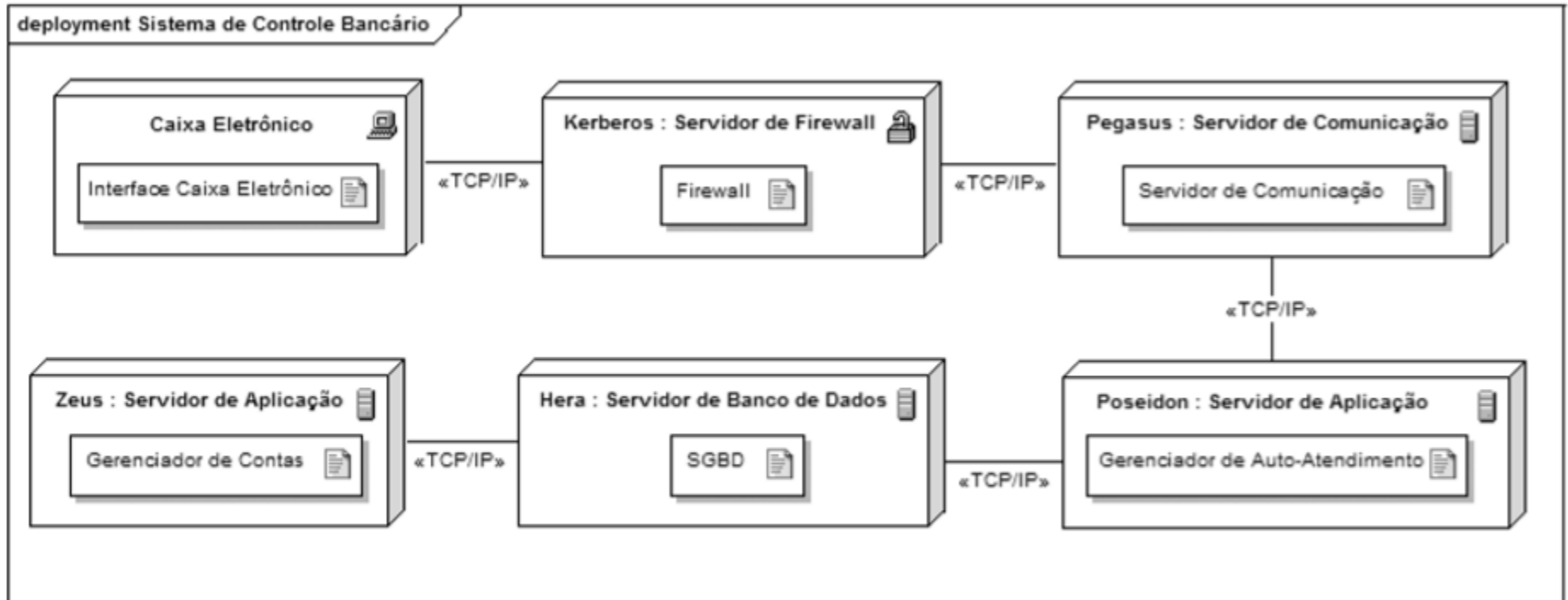


Figura 1.11 – Exemplo de Diagrama de Implantação.

# Diagramas Dinâmicos: Diagrama de Sequência

---

**O diagrama de sequência é um diagrama comportamental que preocupa-se com a ordem temporal em que as mensagens são trocadas entre os objetos envolvidos em um determinado processo para a realização de uma operação.**

# Diagramas Dinâmicos: Diagrama de Sequência

---

O diagrama de sequência baseia-se **em um caso de uso definido pelo diagrama de mesmo nome** e apoia-se no diagrama de classes para determinar os objetos das classes envolvidas em um processo.

# Diagramas Dinâmicos: Diagrama de Sequência

---

Em um diagrama de sequência, os seguintes elementos podem ser encontrados:

- **Linhas verticais** representando o tempo de vida de um objeto (*lifeline*);

Estas linhas verticais são preenchidas por barras verticais que indicam exatamente quando um objeto passou a existir.

Quando um objeto desaparece, existe um "X" na parte inferior da barra;

# Diagramas Dinâmicos: Diagrama de Sequência

---

Em um diagrama de sequência, os seguintes elementos podem ser encontrados:

- **Linhas horizontais ou diagonais** representando mensagens trocadas entre objetos.

Estas linhas são acompanhadas de um rótulo que contém o nome da mensagem e, opcionalmente, os parâmetros da mesma.

Observe que também podem existir mensagens enviadas para o mesmo objeto, representando uma iteração.



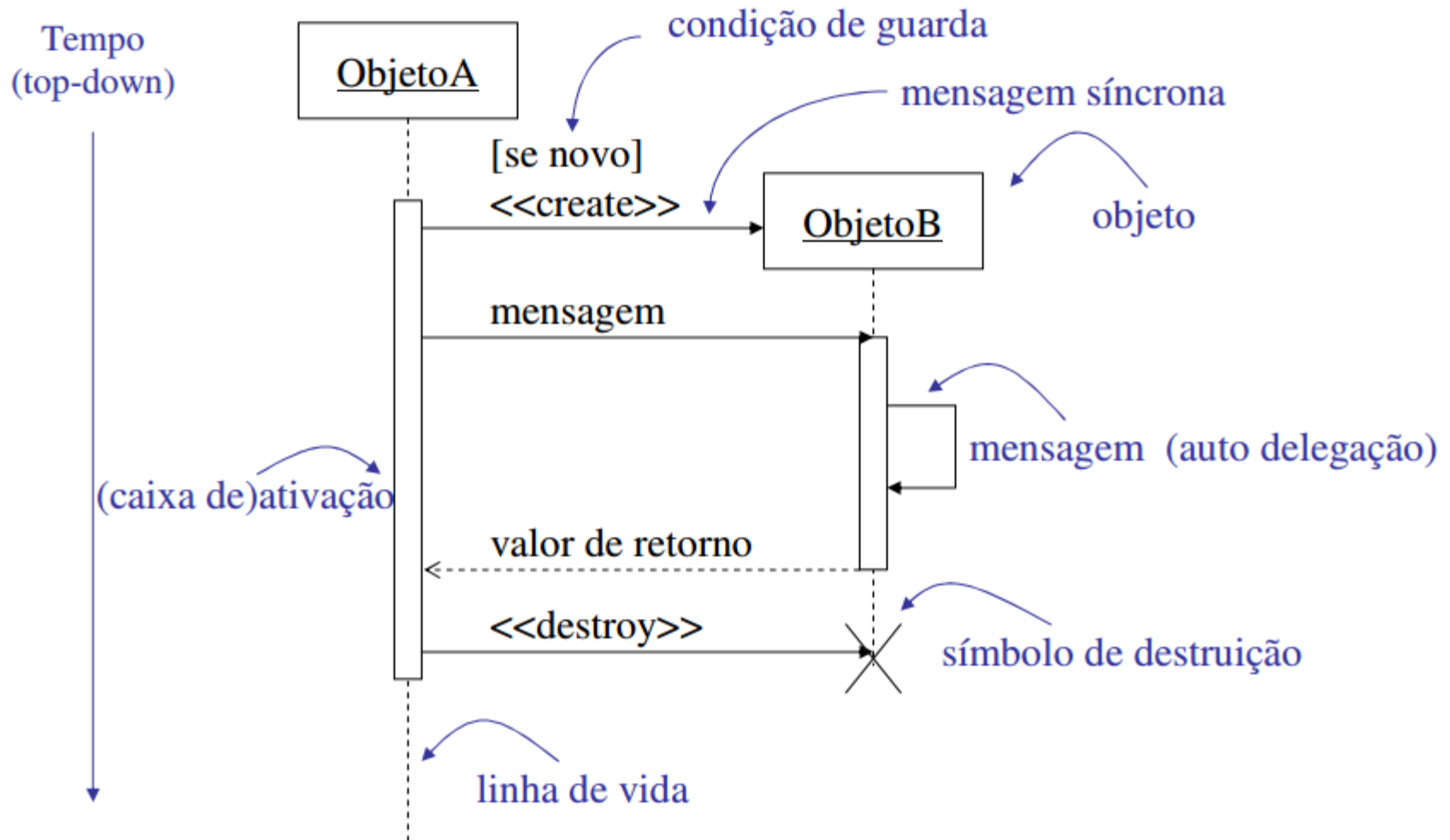
# Diagramas Dinâmicos: Diagrama de Sequência

---

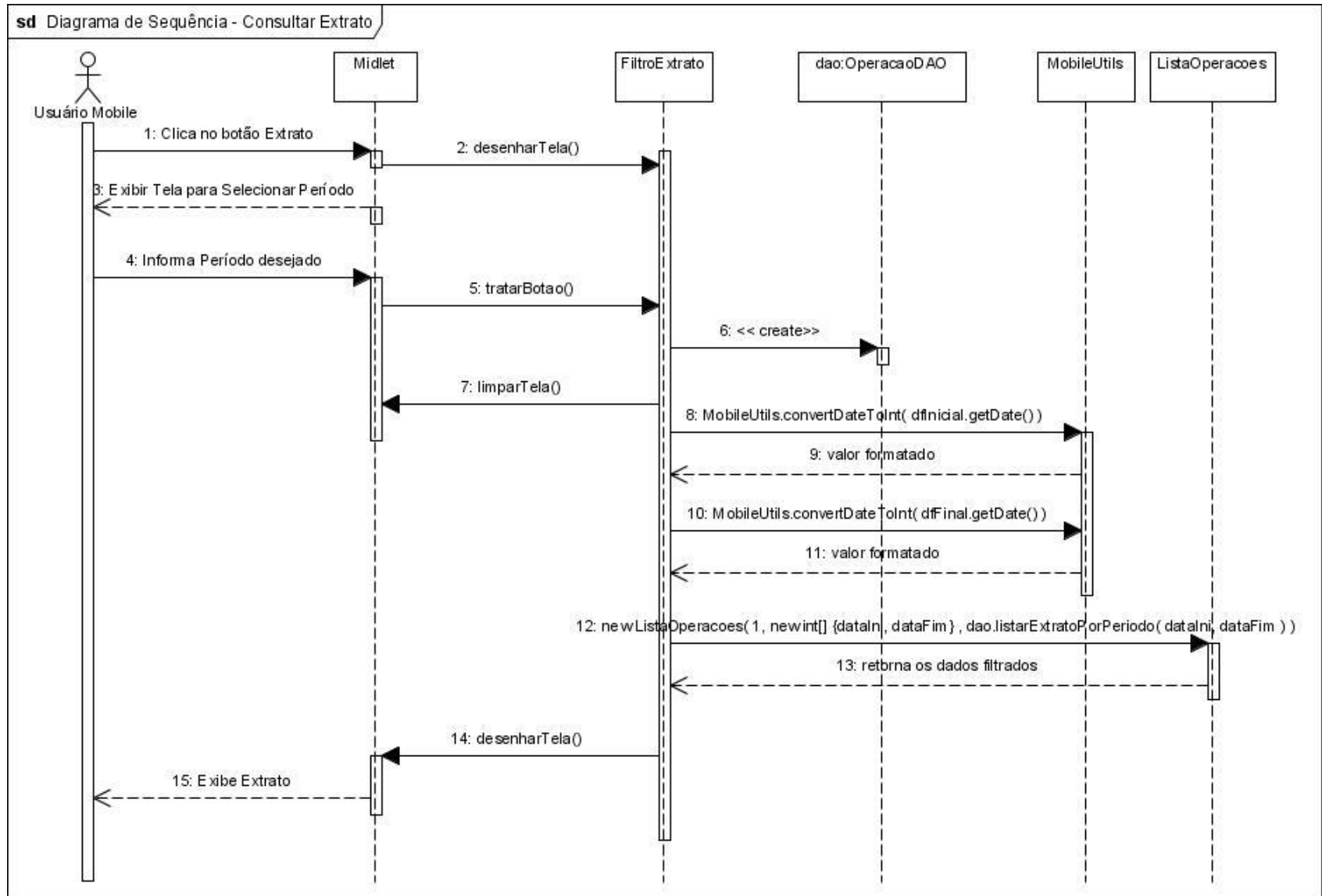
Em um diagrama de sequência, os seguintes elementos podem ser encontrados:

- **Uma condição** é representada por uma mensagem cujo rótulo é envolvido por colchetes;
- **Mensagens de retorno** são representadas por linhas horizontais tracejadas. Este tipo de mensagem não é frequentemente representada nos diagramas, muitas vezes porque sua utilização leva a um grande número de setas no diagrama, atrapalhando o entendimento do mesmo. Este tipo de mensagem só deve ser mostrada quando for fundamental para a clareza do diagrama.

# Diagramas Dinâmicos: Diagrama de Sequência



# Diagramas Dinâmicos: Diagrama de Sequência



# Diagramas Dinâmicos: Diagramas de colaborações

---

O diagrama de comunicação era conhecido como de colaboração até a versão 1.5 da UML, tendo seu nome modificado para **diagrama de comunicação** a partir da versão 2.0.

Está amplamente associado ao diagrama de sequência.

As informações mostradas no diagrama de comunicação com frequência são praticamente as mesmas apresentadas no de sequência, porém com um enfoque distinto, visto que esse diagrama não se preocupa com a temporalidade do processo, concentrando-se em como os elementos do diagrama estão vinculados e quais mensagens trocam entre si durante o processo.

# Diagramas Dinâmicos: Diagramas de colaborações

## Exemplo

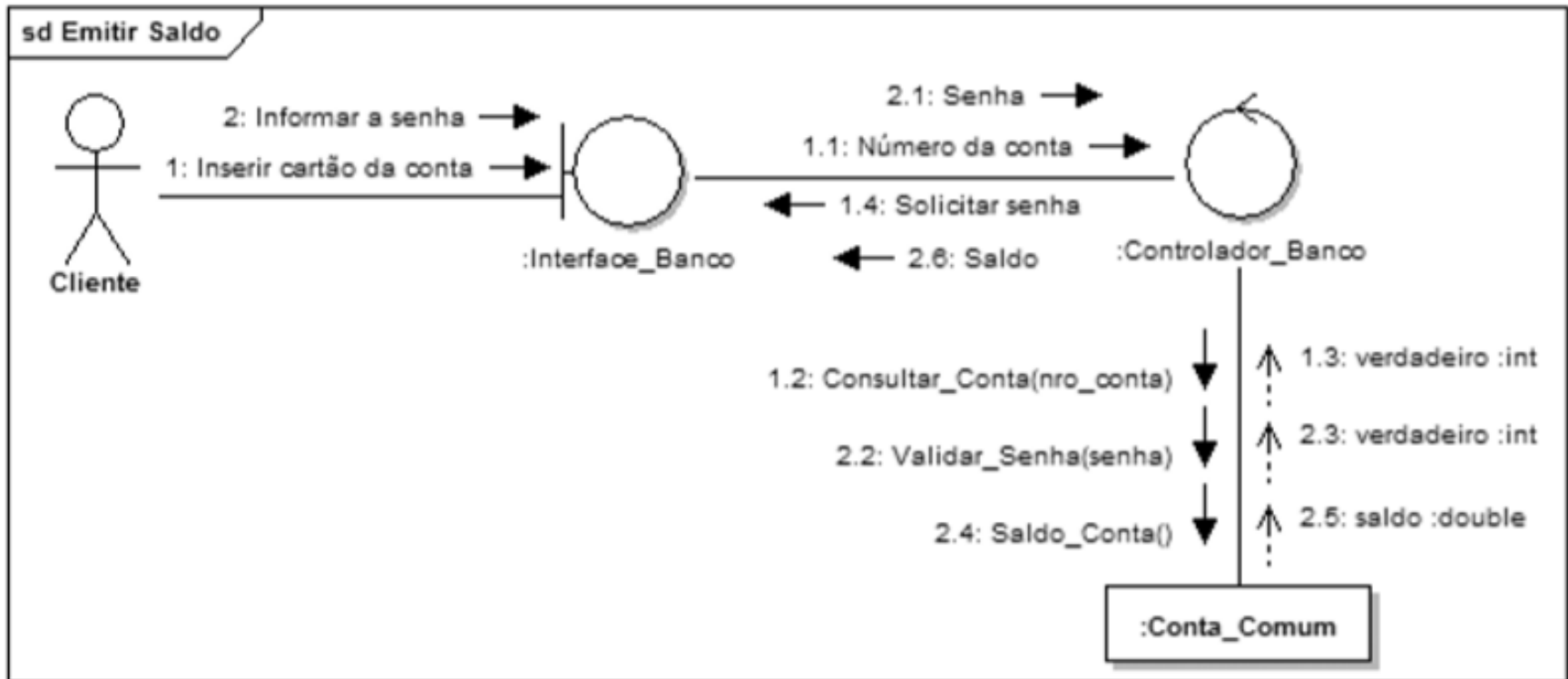


Figura 1.6 – Exemplo de Diagrama de Comunicação.

# Diagramas Dinâmicos: Diagrama de Estados

---

Em um **diagrama de estado**, um objeto possui um comportamento e um estado.

O estado de um objeto depende da atividade na qual ele está processando.

Um diagrama de estado **mostra os possíveis estados de um objeto e as transações** responsáveis pelas suas mudanças de estado.

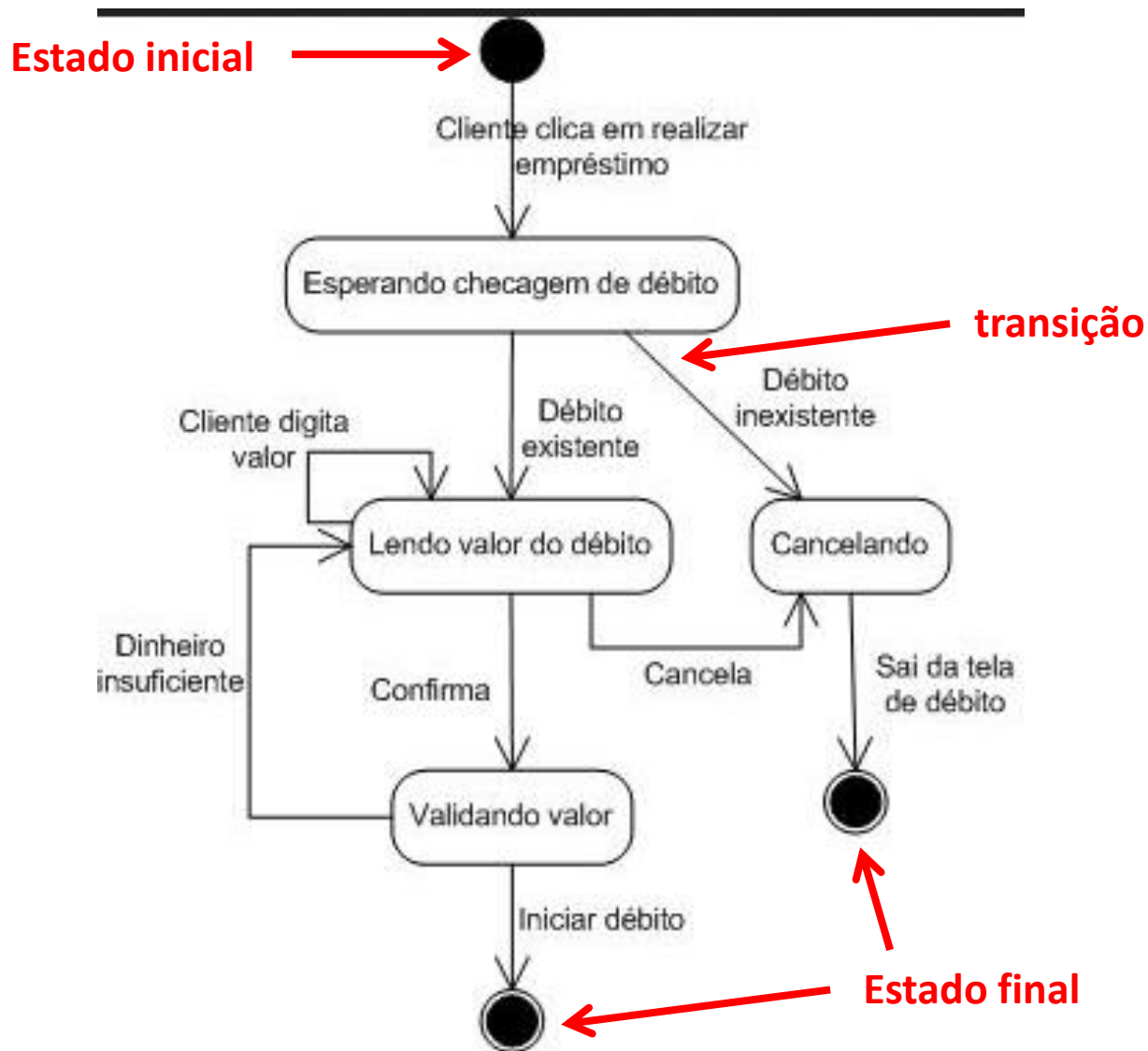
# Diagramas Dinâmicos: Diagrama de Estados

---

Como o diagrama de sequência, o de máquina de estados pode basear-se em um caso de uso, mas também pode ser utilizado para acompanhar os estados de outros elementos, como, por exemplo, uma instância de uma classe.

Além de poder ser utilizado para expressar o comportamento de uma parte do sistema, quando é chamado de máquina de estado comportamental.

# Diagramas Dinâmicos: Diagrama de Estados





# Diagramas Dinâmicos: Diagramas de Atividade

---

O diagrama de atividade era considerado um caso especial do antigo diagrama de gráfico de estados, hoje conhecido como diagrama de máquina de estados.

A partir da UML 2.0, foi considerado independente do diagrama de máquina de estados.

# Diagramas Dinâmicos: Diagramas de Atividade

---

O diagrama de atividade **preocupa-se em descrever os passos a serem percorridos para a conclusão de uma atividade específica**, podendo esta ser representada por um método com certo grau de complexidade, um algoritmo, ou mesmo por um processo completo.

O diagrama de atividade concentra-se na representação do fluxo de controle de uma atividade.

# Diagramas Dinâmicos: Diagramas de Atividade

---

**Diagrama de atividades ilustra o fluxo de controle entre atividades, enquanto que um diagrama de estados ilustra o fluxo de controle entre estados.**

# Diagramas Dinâmicos: Diagramas de Atividade

## Exemplo

### 3.14.5 Documentação do Caso de Uso Emitir Saldo

Tabela 3.6 – Documentação do Caso de Uso Emitir Saldo

Nome do Caso de Uso	Emitir Saldo
Caso de Uso Geral	
Ator Principal	Cliente
Atores Secundários	
Resumo	Descreve os passos necessários para um cliente obter o saldo referente a uma determinada conta
Pré-Condições	
Pós-Condições	
Fluxo Principal	
Ações do Ator	Ações do Sistema
1. Informar o número da conta	
	2. Verificar a existência da conta
	3. Solicitar a senha da conta
4. Informar a senha	
	5. Verificar se a senha está correta
	6. Emitir o saldo
Restrições/Validações	1. A conta precisa existir e estar ativa 2. A senha precisa estar correta
Fluxo de Exceção I – Conta não encontrada	
Ações do Ator	Ações do Sistema
	1. Comunicar ao cliente que o número da conta informada não foi encontrado
Fluxo de Exceção II – Senha inválida	
Ações do Ator	Ações do Sistema
	1. Comunicar ao cliente que a senha fornecida não combina com a senha da conta

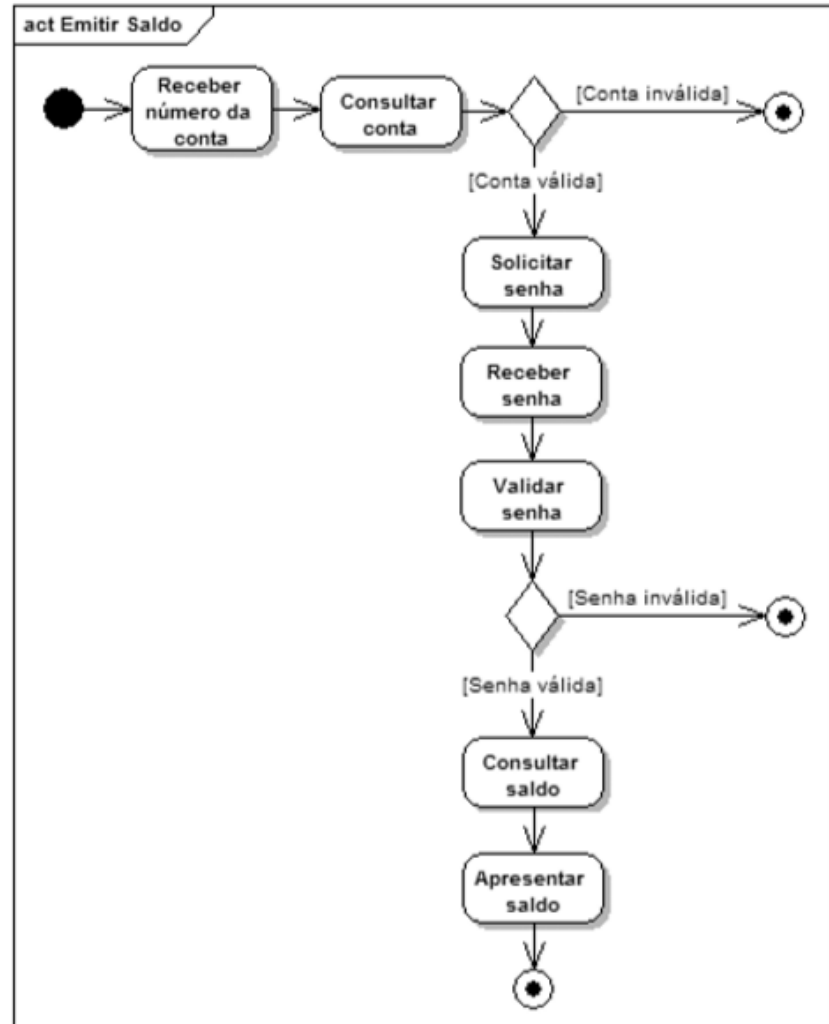


Figura 1.8 – Exemplo de Diagrama de Atividade.

# Diagramas Dinâmicos: Diagramas de Atividade

---

## Elementos mais utilizados do diagrama de atividades

**Activity** – É a atividade propriamente dita. Usamos este elemento quando citamos uma atividade no diagrama.

Por exemplo: “Processar Pedido” é uma atividade que seria ilustrada com esta forma.

**Partition** – É comum chamarmos de “Raia”, fazendo uma analogia com as raia de uma piscina.

Podem ser representadas na vertical ou na horizontal. Ilustram fronteiras entre módulos, funcionalidades, sistemas ou sub-sistemas, conforme o nível de detalhe e foco do diagrama.

# Diagramas Dinâmicos: Diagramas de Atividade

---

## Elementos mais utilizados do diagrama de atividades

**Decision** – Representa uma decisão que pode desviar o fluxo ilustrado no diagrama.

Utilizado quando lidamos com condições. Por exemplo:

“Pagamento aprovado?”

Se sim, desvia para a atividade “**Gerar Boleto**”,

Se não, vai para atividade “**Pagar novamente**”.

**Initial** – É o primeiro elemento do diagrama. Define o início do fluxo.

Um diagrama de atividades pode ter mais de um elemento deste, pois seu início pode ser dar em mais de um “local”.

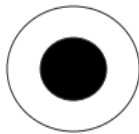


# Diagramas Dinâmicos: Diagramas de Atividade

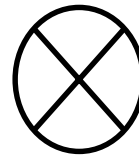
---

## Elementos mais utilizados do diagrama de atividades

**Final** – É o último elemento do diagrama. Define o fim do fluxo. Um diagrama de atividades pode ter mais de um elemento deste também, pois o fim do fluxo pode ocorrer em várias partes do diagrama. O ideal é utilizar o elemento “Flow Final”, mas é um conceito mais avançado.



Estado Final  
(conclusão)



Flow Final

# Diagramas Dinâmicos: Diagramas de Atividade

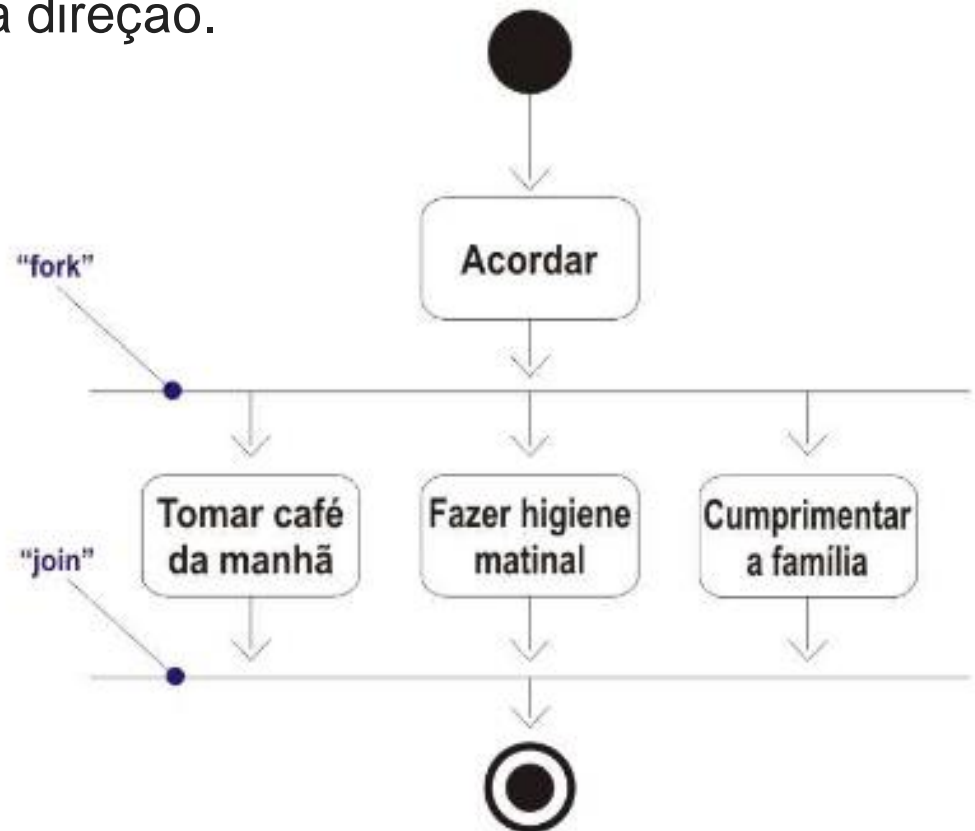
## Elementos mais utilizados do diagrama de atividades

**Fork** – tem como finalidade dividir o fluxo em mais de uma direção

**Join** – tem finalidade inversa, ou seja, faz a união de várias direções do fluxo em uma única direção.

Um “**Fork**” tem uma transição de entrada e diversas transições de saída (ao menos duas).

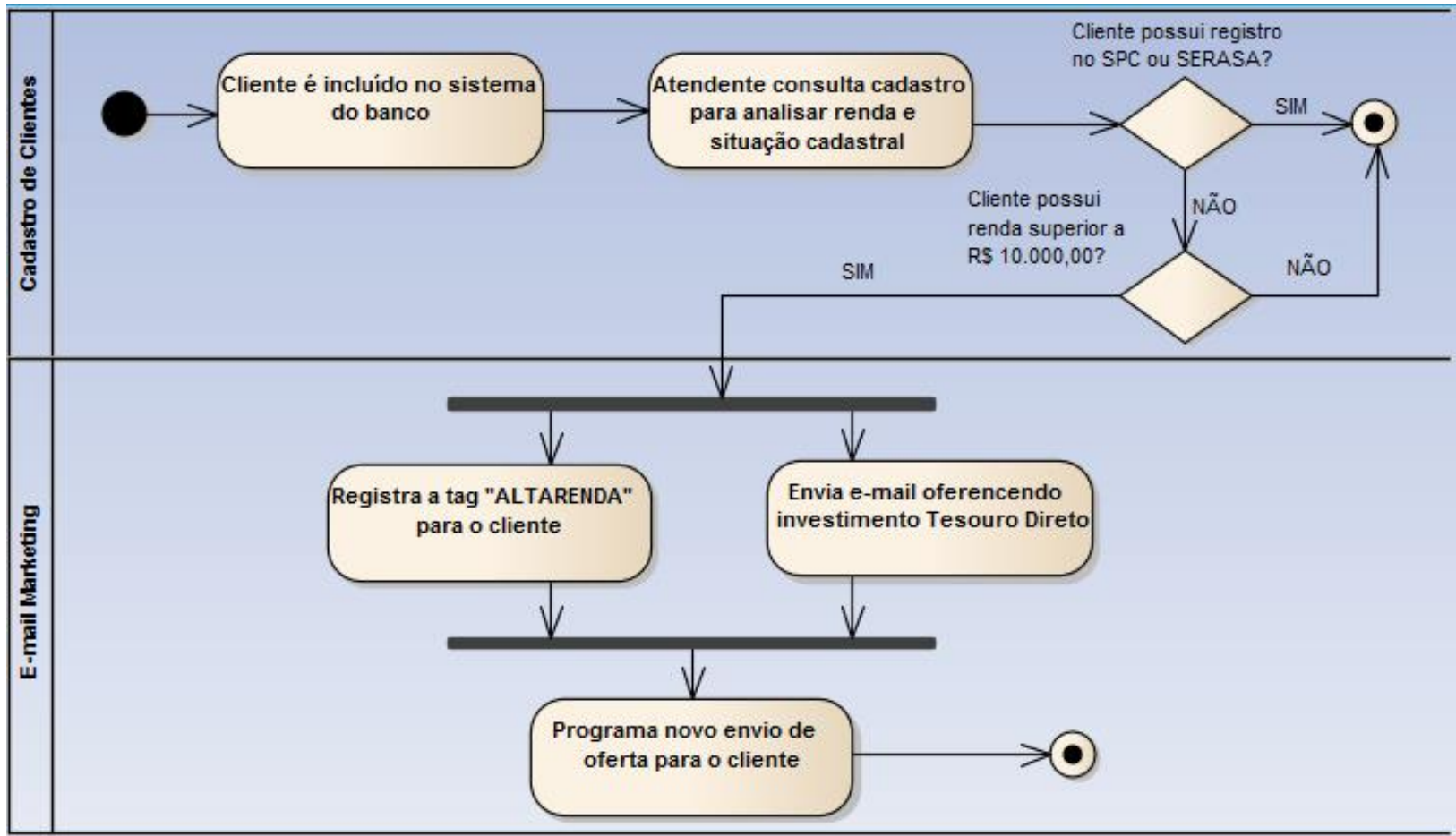
Com “**Join**” a transição seguinte só ocorre quando todos os estados nas transições de entrada tenham completado suas atividades passando assim ao próximo estágio do fluxo.





# Diagramas Dinâmicos: Diagramas de Atividade

## Exemplo



# Diagramas Dinâmicos: Diagramas de Atividade

---

**Fluxogramas normalmente são limitados a processos sequenciais enquanto Diagramas de atividades podem manipular processos paralelos.**

A característica de paralelismo é importante na modelagem de negócios pois nem sempre os procedimentos se caracterizam por uma sequencia mecânica de passos.

# Diagramas Dinâmicos: Diagramas de Atividade

---

Activity Diagram0 / Activity Diagram



1 2 3 4 5 6 7 8 9 10 11

1. Raia vertical
2. Raia horizontal
3. Estado inicial
4. Atividade
5. Atividades que pode ser descomposta em varias atividades
6. Estado final
7. Flow final
8. Sentido da atividade
9. Nodo de decisão
10. Fork
11. Join



## Bibliografia

- Boock, G. and Rumbaugh, J. The Unified Modeling Language User Guide . Addison-Wesley, 1999
- Arlow, J. and Neustadt, I. UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design, 2nd Edition, The Addison-Wesley Object Technology Series, 2005.
- Rumbaugh, J.; Jacobson, I. and Booch , G. The Unified Modeling Language Reference Manual, 2nd Edition, The Addison-Wesley Object Technology Series, 2004.
- Boock, G.; Rumbaugh, J. and Jacobson, I; Unified Modeling Language User Guide, 2nd Edition, The Addison-Wesley Object Technology Series, 2005.
- Jacobson, I; Boock, G. and Rumbaugh, J., Unified Software Development Process, Addison-Wesley, Janeiro 1999.
- Larman, C. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design Prentice-Hall, New Jersey - USA, 1997
- Bezerra, E. Princípios de Análise e Projeto com a UML, ed. Campus-Elsevier. 2003.