

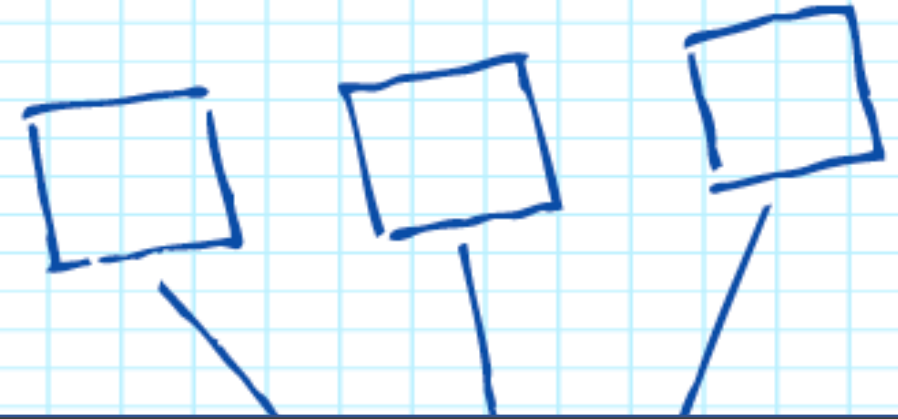


# Engenharia de Software I

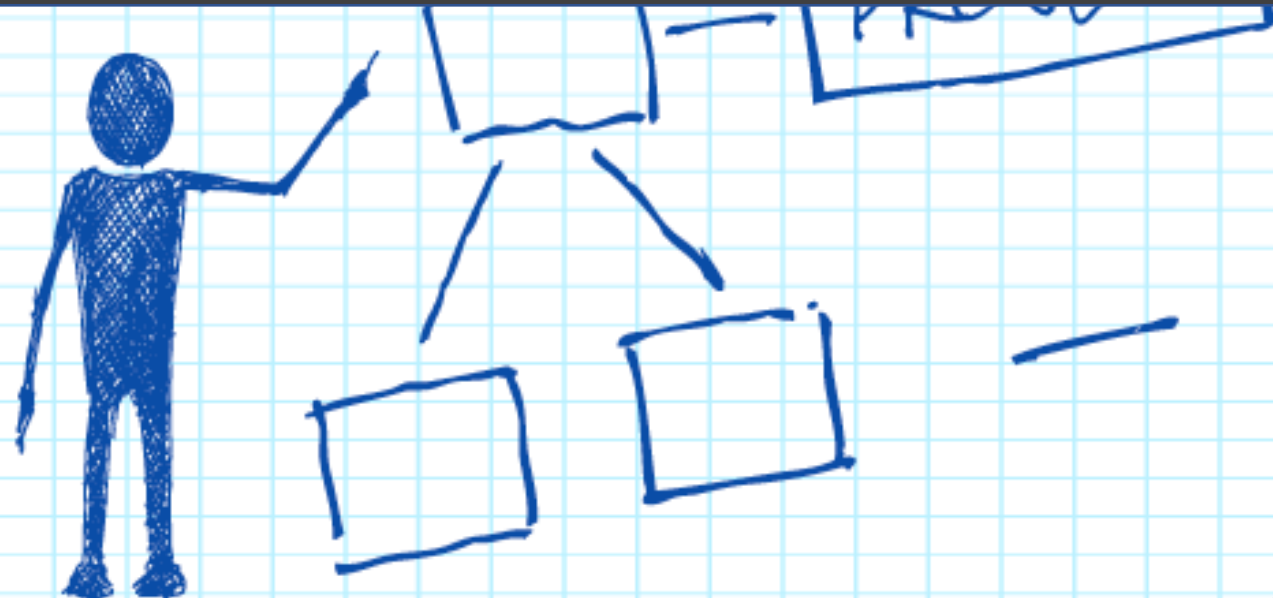
**Prof<sup>a</sup>. Me. Cynara Leão Garcia**

**[cynara.garcia@unicesumar.edu.br](mailto:cynara.garcia@unicesumar.edu.br)**

# INTRODUÇÃO A UML

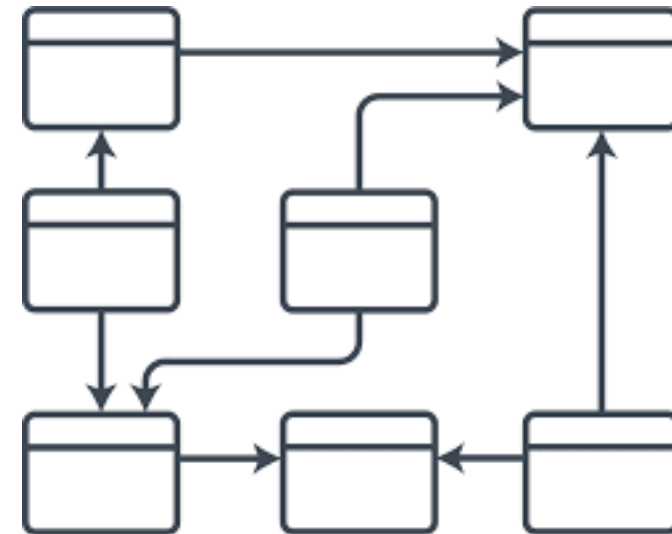


## Introdução a UML



# A Linguagem UML (*Unified Modeling Language*)

- UML: Linguagem de Modelagem Unificada
- É uma notação gráfica (visual)
  - Não é uma linguagem de programação
  - Modela sistemas orientados a objetos
- Define diagramas padronizados
- É extensível
- É complexa (muitos diagramas)

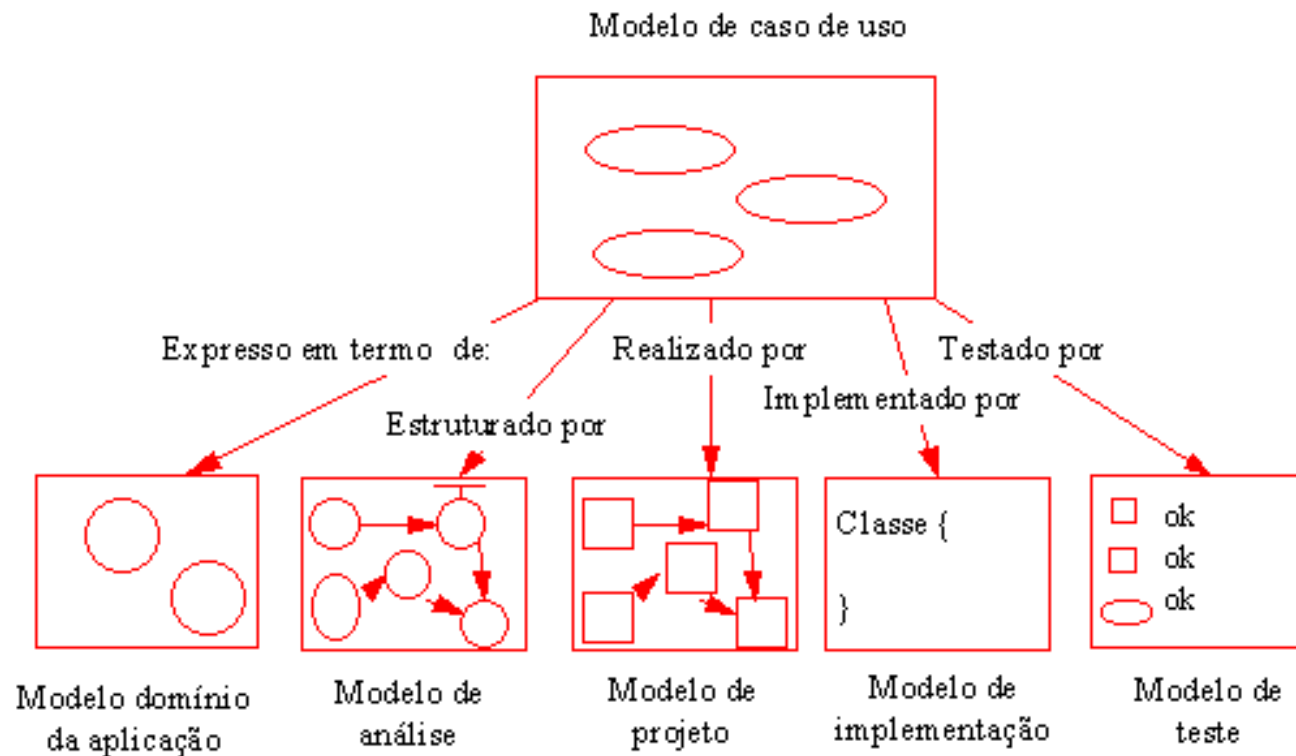


# De onde surgiu?

- Da união de três técnicas de modelagem
  - **Método de Booch – Grady Booch**
    - Análise de requisitos, que permite o estabelecimento das operações fundamentais do sistema;
    - Análise do domínio, que permite a construção da estrutura lógica referente ao domínio em questão;
    - Desenho, que permite a construção da estrutura física do sistema, fazendo o mapeamento da estrutura lógica anteriormente construída, conduzindo, assim, à construção de protótipos.

# De onde surgiu?

- Da união de três técnicas de modelagem
  - **Método de OOSE – Ivair Jacobson**



# De onde surgiu?

- Da união de três técnicas de modelagem
  - **Método de OMT – James Rumbaugh**
    - O analista identifica classes e objetos;
    - Constrói um dicionário de dados;
    - Adiciona associações entre classes;
    - Adiciona atributos a objetos,
    - Organiza classes através de heranças,
    - Testa passos de acesso usando cenários,
    - Refina o modelo e agrupa classes em módulos.

# De onde surgiu?

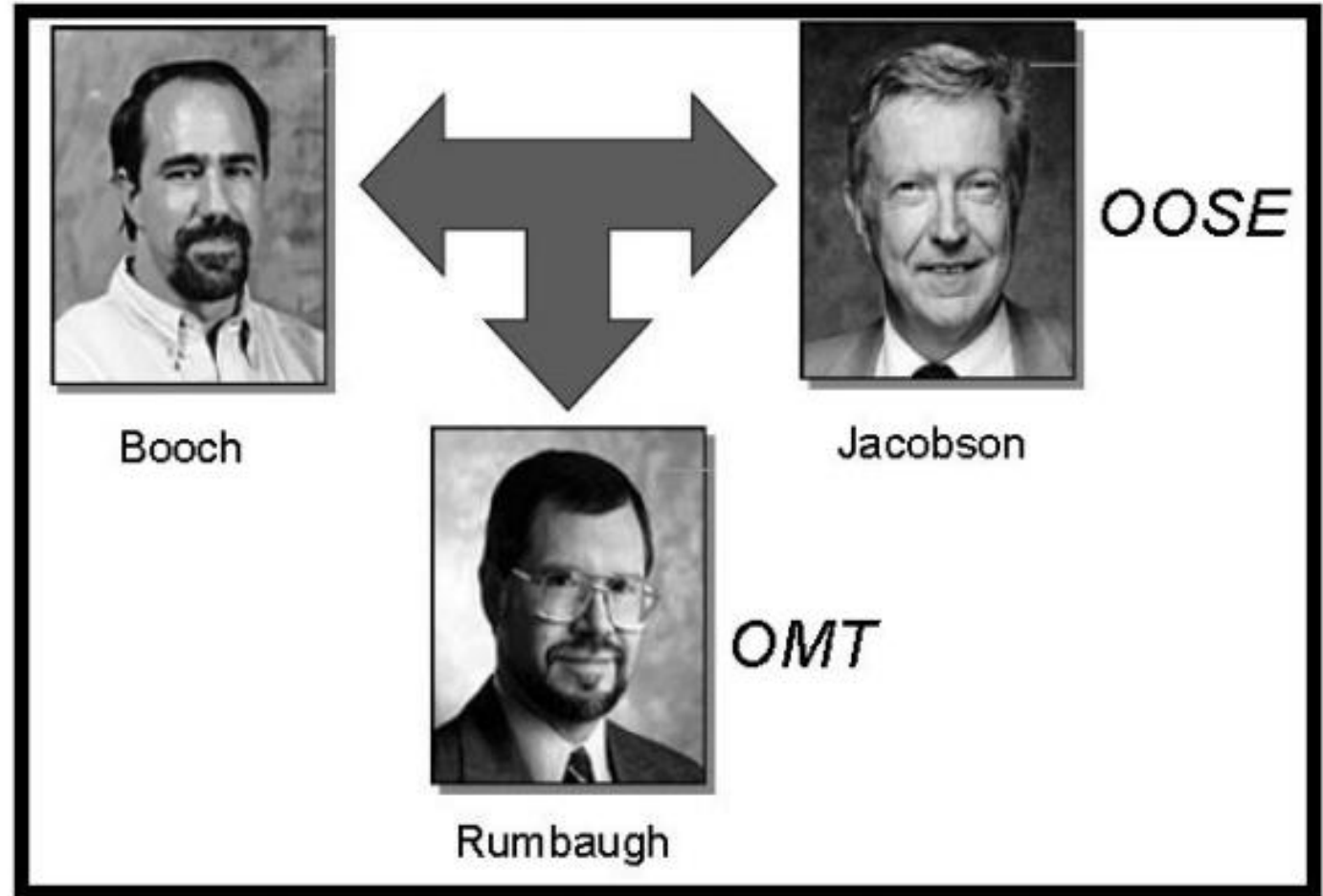
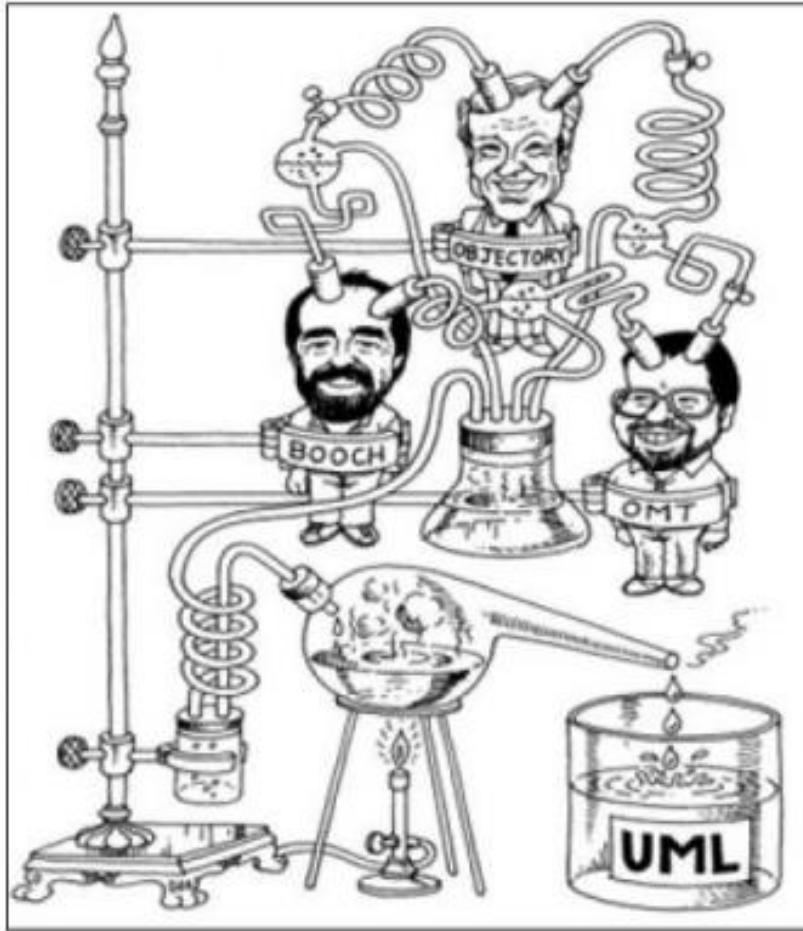
- Os amigos começaram então a unifica-las em meados da década de noventa.

Anos 90





# Fundadores da UML





# História da UML

- 1994: Booch, Jacobson e Rumbaugh começaram a unificar suas notações;
- 1996: Primeira versão (beta) da UML foi liberada
- 1996/1997: Grandes empresas formaram a “UML Partners”
  - HP, IBM, Microsoft, Oracle, etc.
- 1997: UML foi adotada pela OMG (Object Management Group)
  - Linguagem padrão de modelagem



# O que é modelagem?

- Um modelo é uma simplificação da realidade
- Modelagem de software é a atividade de construir modelos do sistema
- A UML pode ser usada em qualquer processo de software
  - Ela é usada principalmente nas atividade de especificação de requisitos e projeto.



# Por que modelar?

- Tão essencial quanto ter uma planta antes da construção de uma casa
  - Melhora a comunicação entre os membros da equipe e o cliente
  - A equipe entende melhor o sistema
  - Permite analisar o sistema sobre vários aspectos
  - Facilita a programação e a manutenção
  - Diminui a possibilidade de erros



# Por que usar UML?

- Bons modelos são essenciais para a comunicação entre os *stakeholders*
- Padronização
  - A equipe entende a modelagem facilitando a manutenção
- Facilita a programação
  - Integração entre ferramentas para modelagem e geração de código



# Modelagem Orientada a Objetos

- Maneira natural de visualizar o software
- Modela o software semelhante ao mundo real – usando objetos
  - Pessoas, animais, plantas, carros, etc.
- Humanos pensam em termos de objetos
  - Mais alto nível



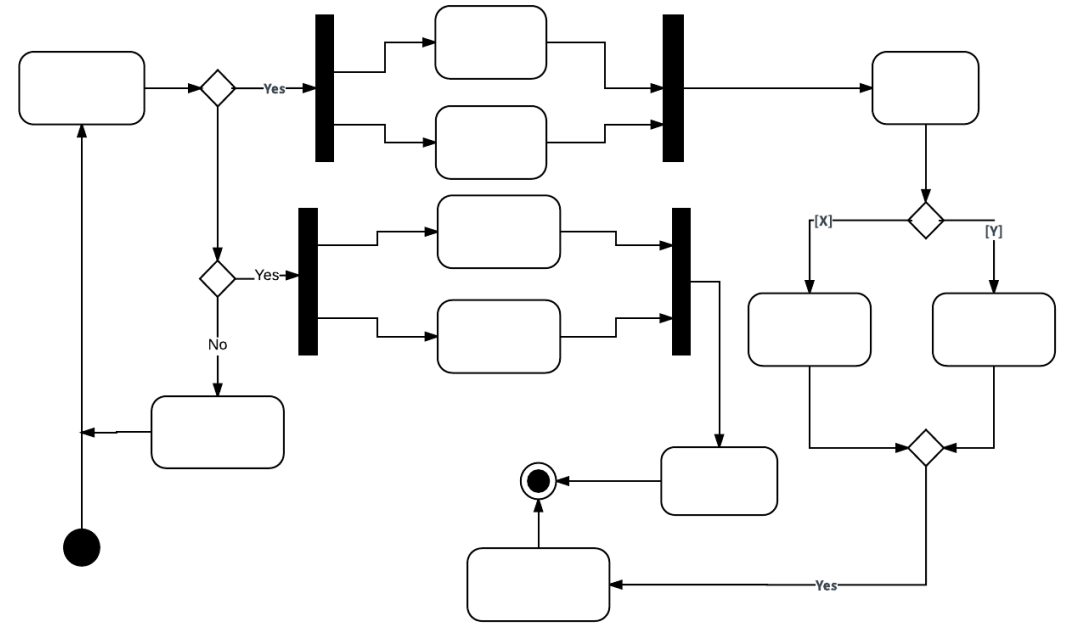
# UML Define 14 Diagramas

- Tipos Principais de Diagramas

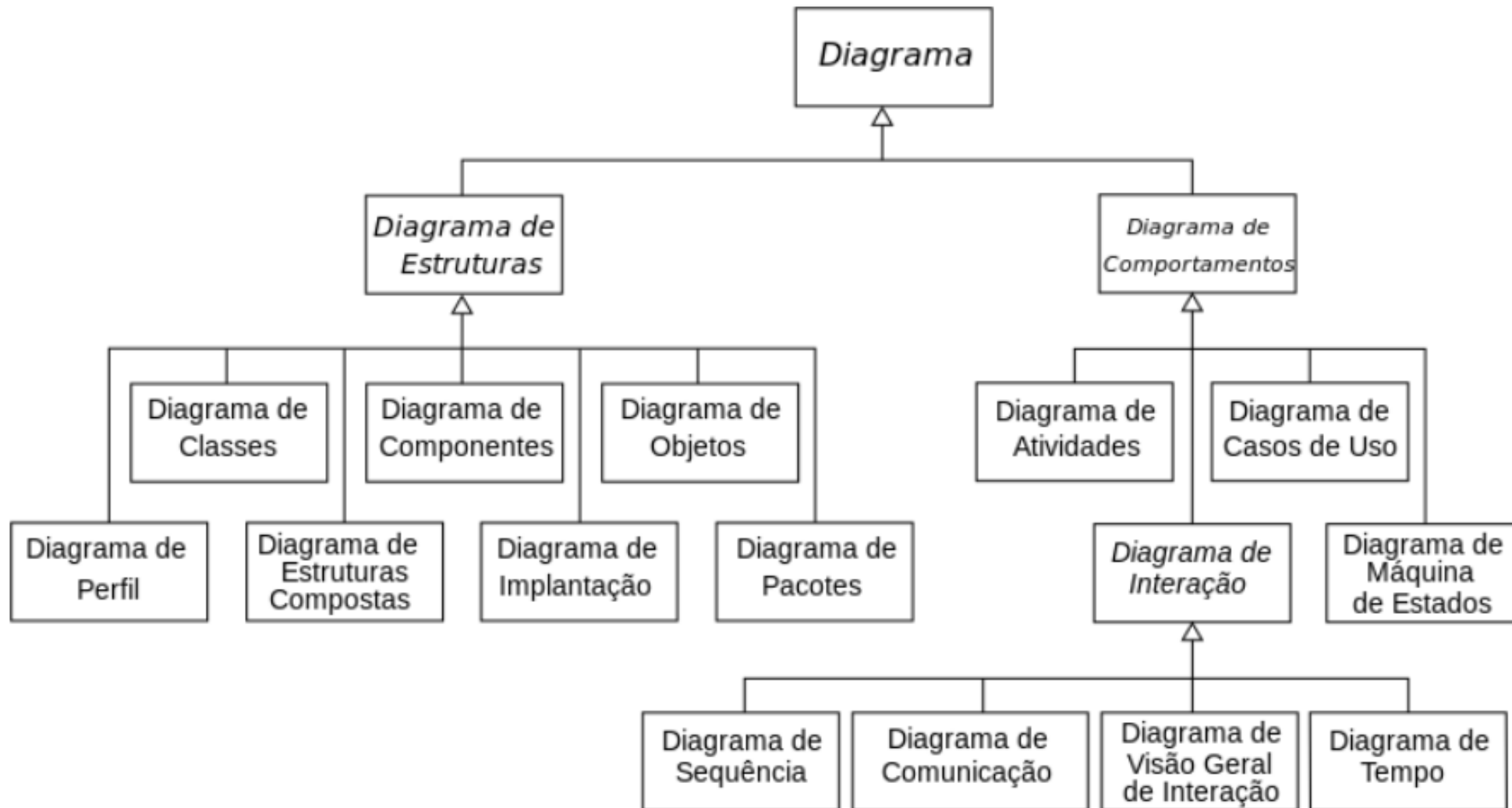
- Estrutural
- Comportamental

- Objetivos

- Visualizar o sistema
- Especificar estrutura e/ou comportamento
- Guiar e documentar as decisões

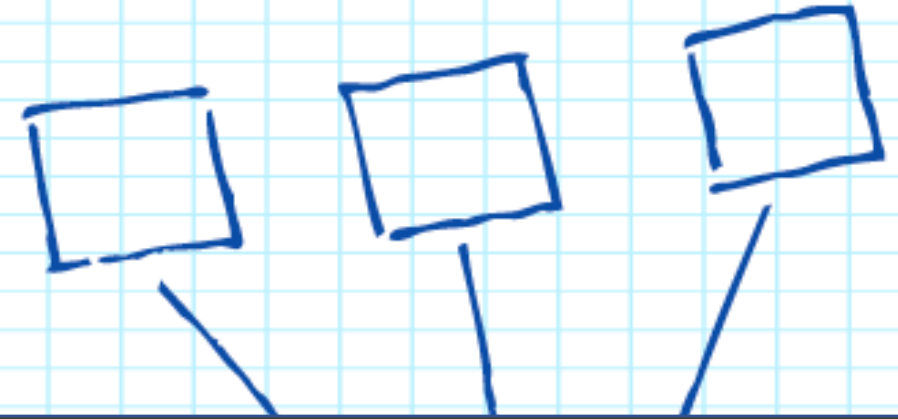


# Classificação dos Diagramas

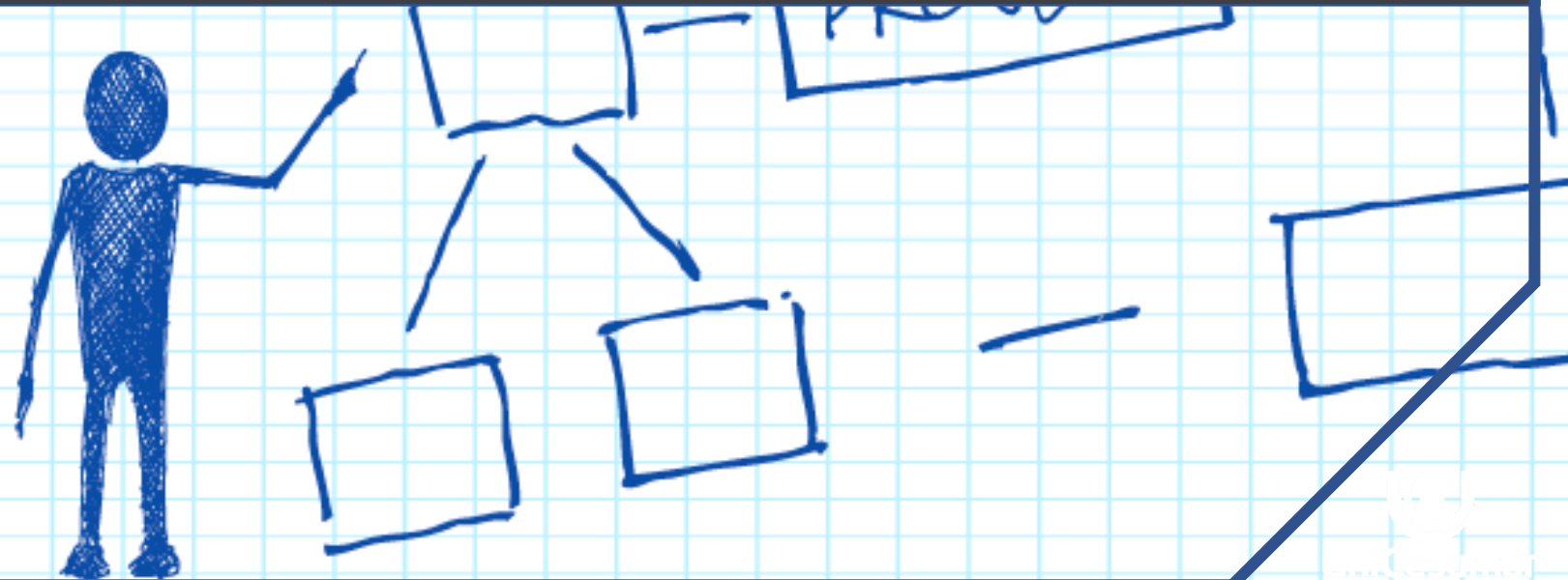




# INTRODUÇÃO A UML

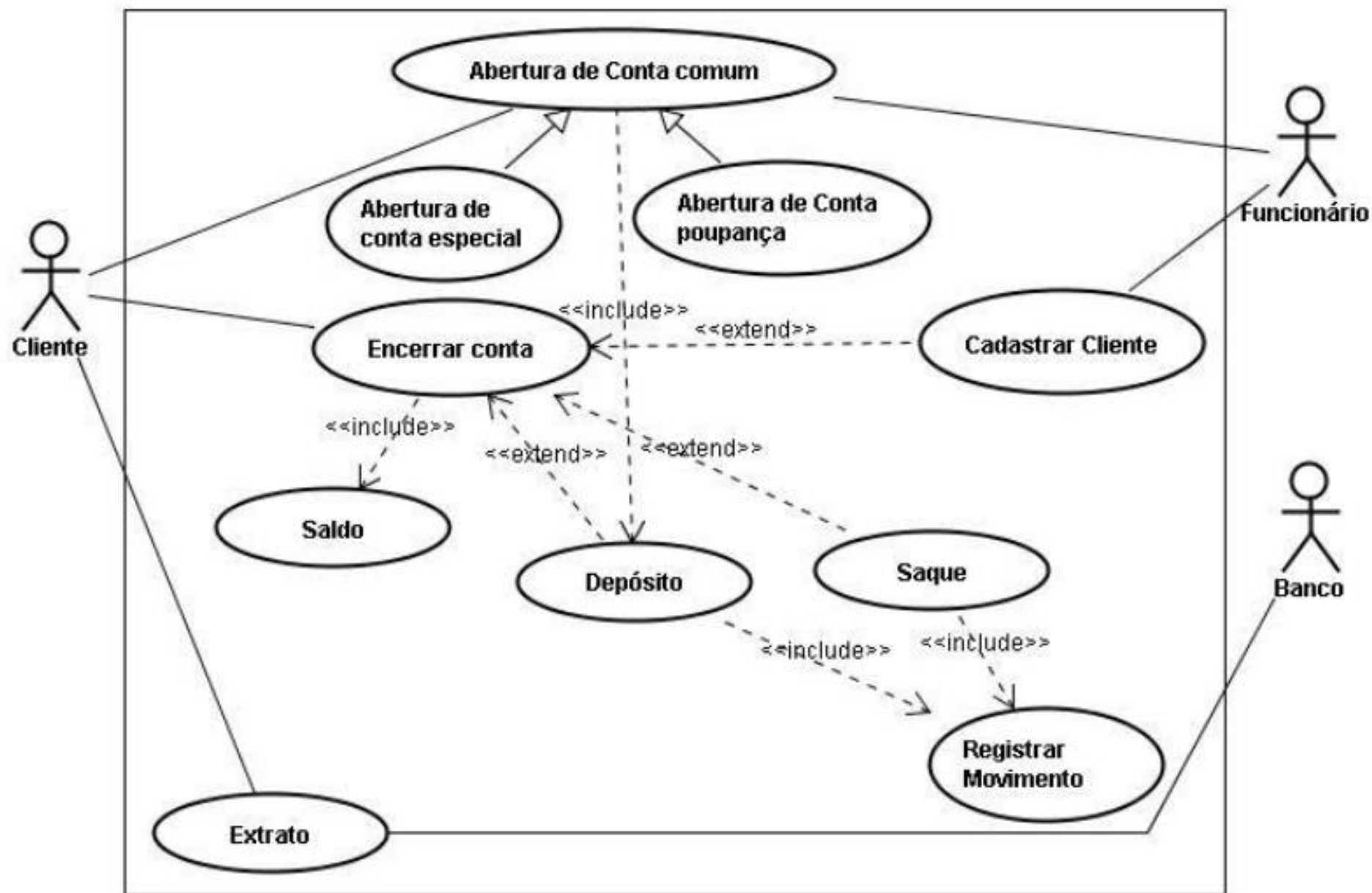


## Resumo dos Diagramas UML



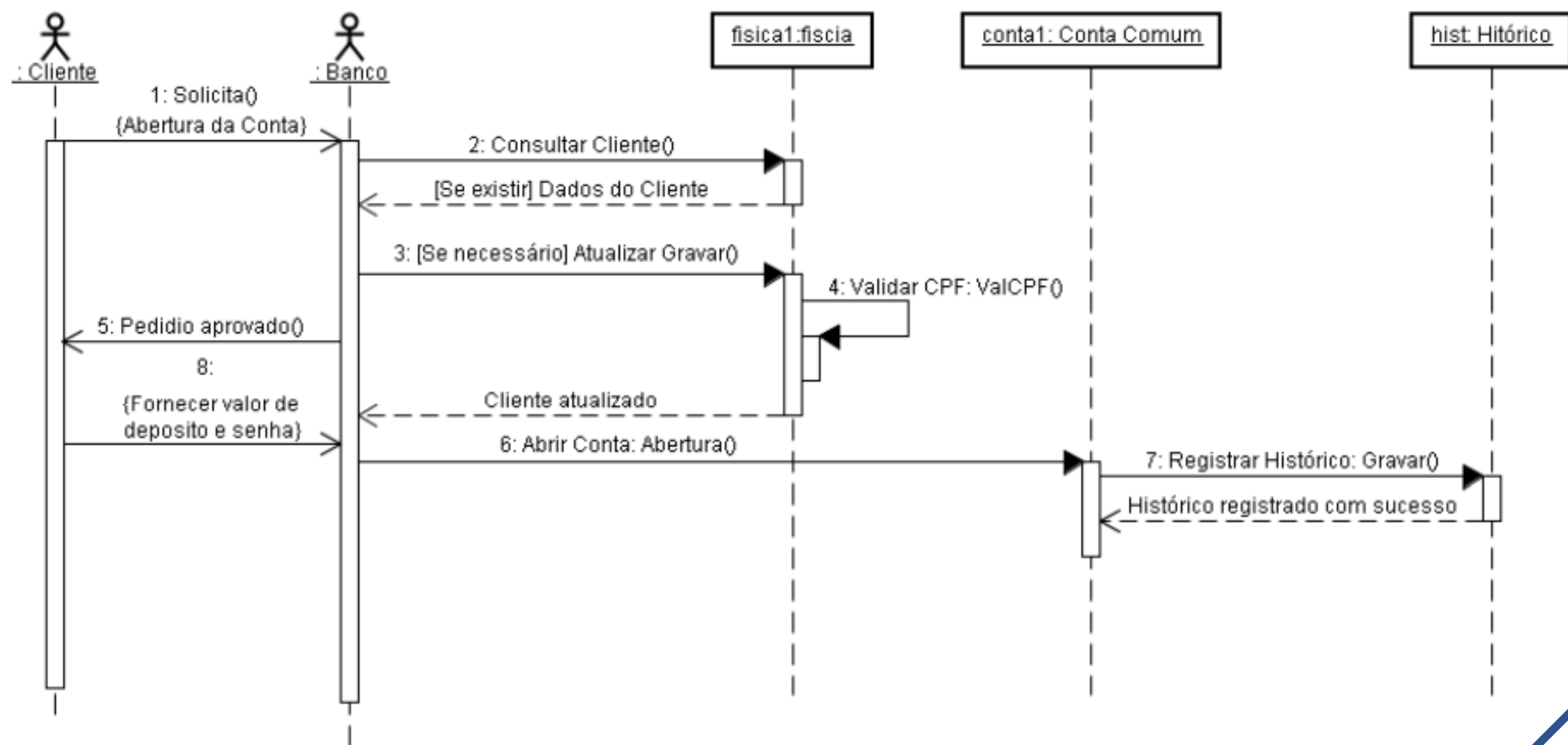
# Diagrama de Caso de Uso

- Diagrama mais geral da UML
- Usado geralmente na fase de especificação de Requisitos
- Mostra
  - Quais os usuários realizam que funcionalidade do sistema
  - Alguns relacionamentos entre essas funcionalidades



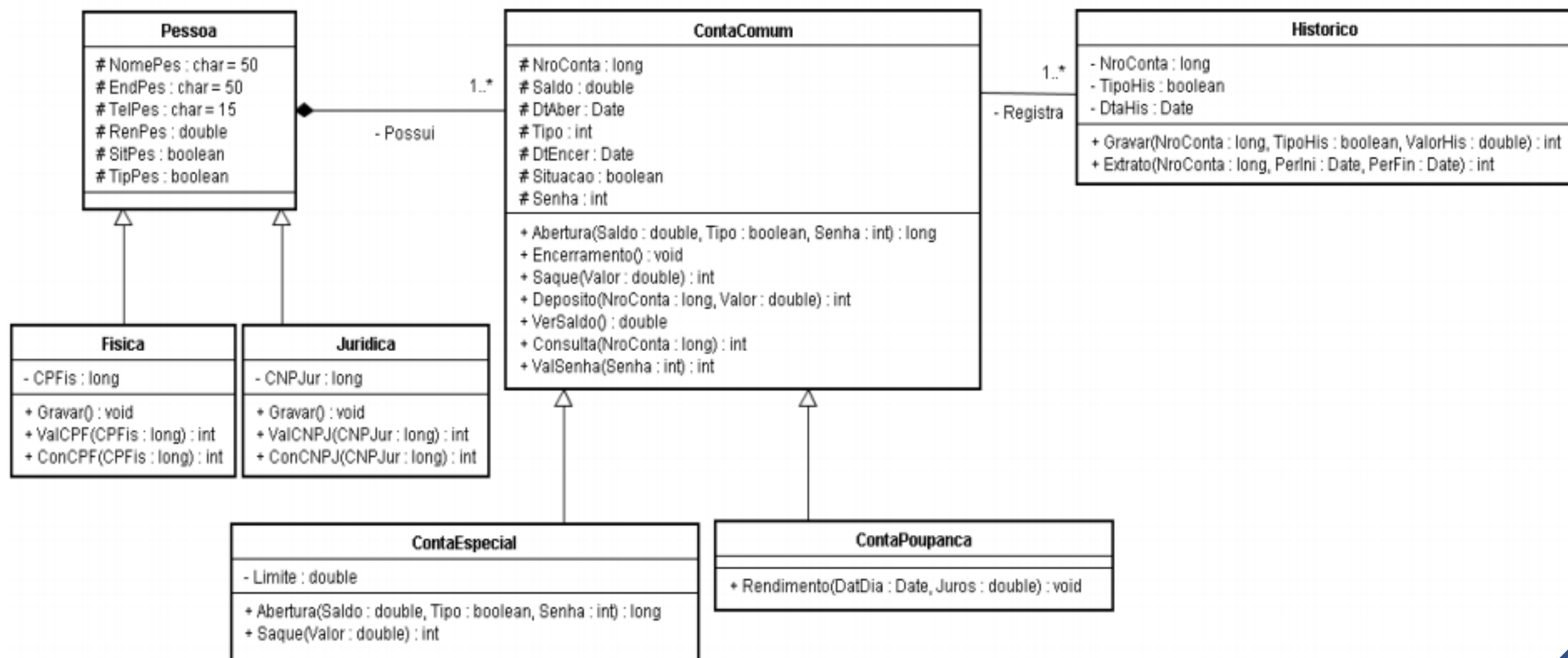
# Diagrama de Sequência

- Preocupa-se com a ordem temporal em que as mensagens são trocadas
- Pode se basear em um Caso de Uso
- Identifica
  - Os eventos associados a funcionalidade modelada
  - O ator responsável por este evento



# Diagrama de Classes

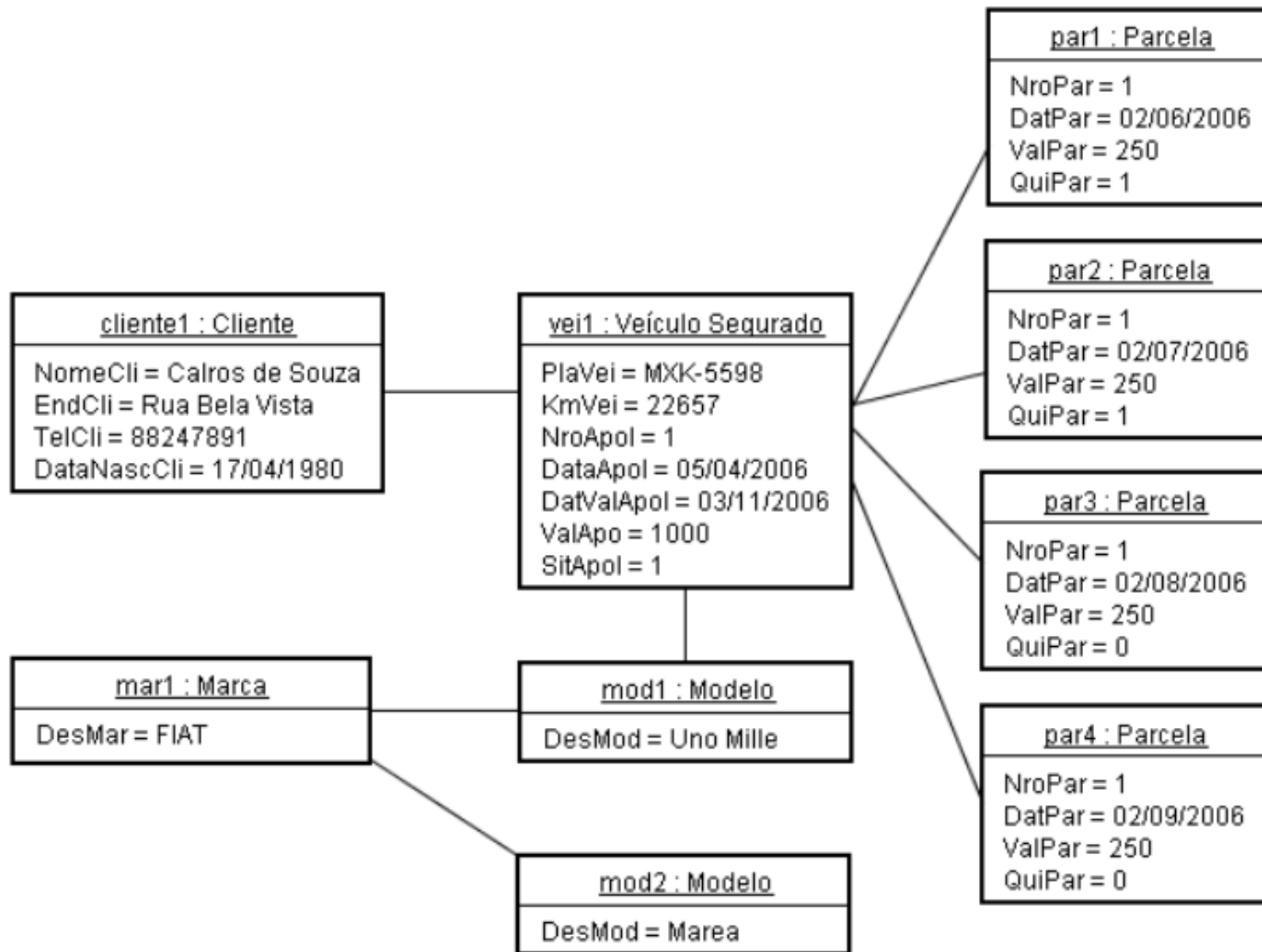
- Diagrama mais utilizado da UML
- Serve de apoio para a maioria dos outros diagramas
- Define a estrutura das classes do sistema
- Estabelece como as classes se relacionam





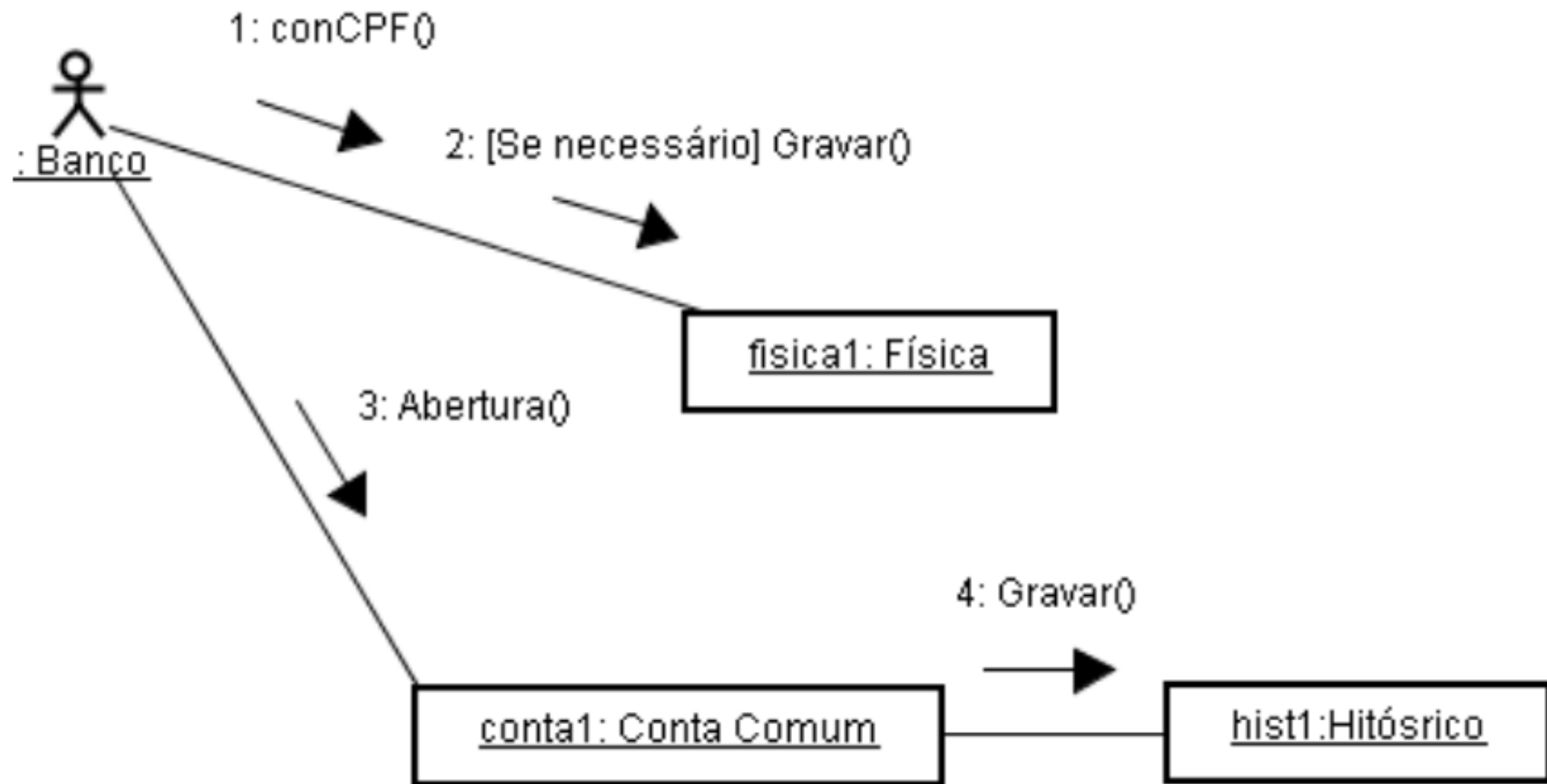
# Diagrama de Objetos

- Complemento do Diagrama de Classes
- Exibe os valores armazenados pelos objetos de um Diagrama de Classes.



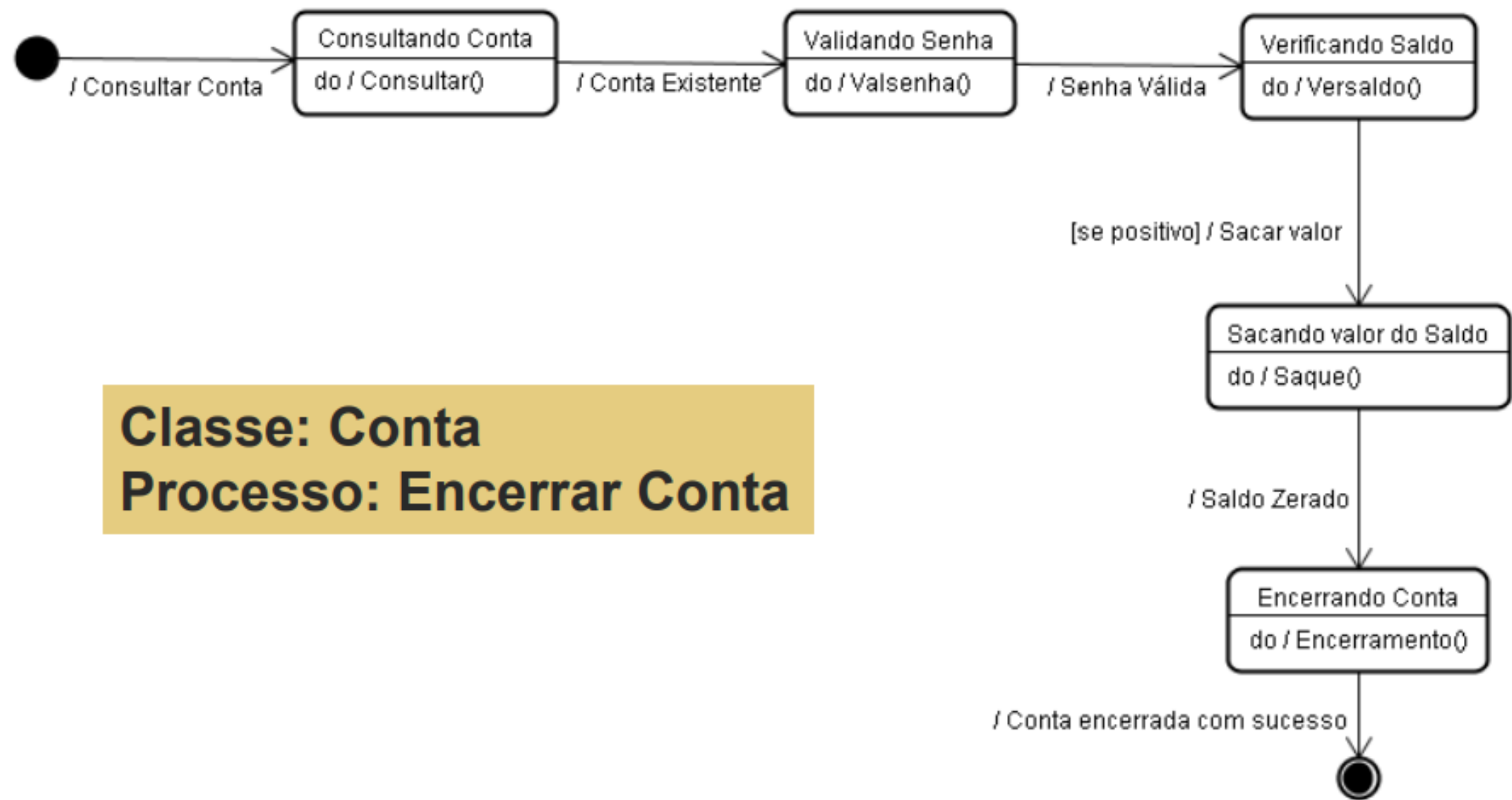
# Diagrama de Comunicação

- Amplamente associado ao Diagrama de Sequência
  - São complementares
- Não se preocupa com a temporalidade
- Define
  - Como os objetos estão vinculados
  - Quais mensagens são trocadas entre os objetos



# Diagrama de Estados

- Modela as mudanças sofridas por um objeto dentro de um determinado processo
- Pode ser utilizado para acompanhar os estados pelo qual passa uma instância de uma classe.



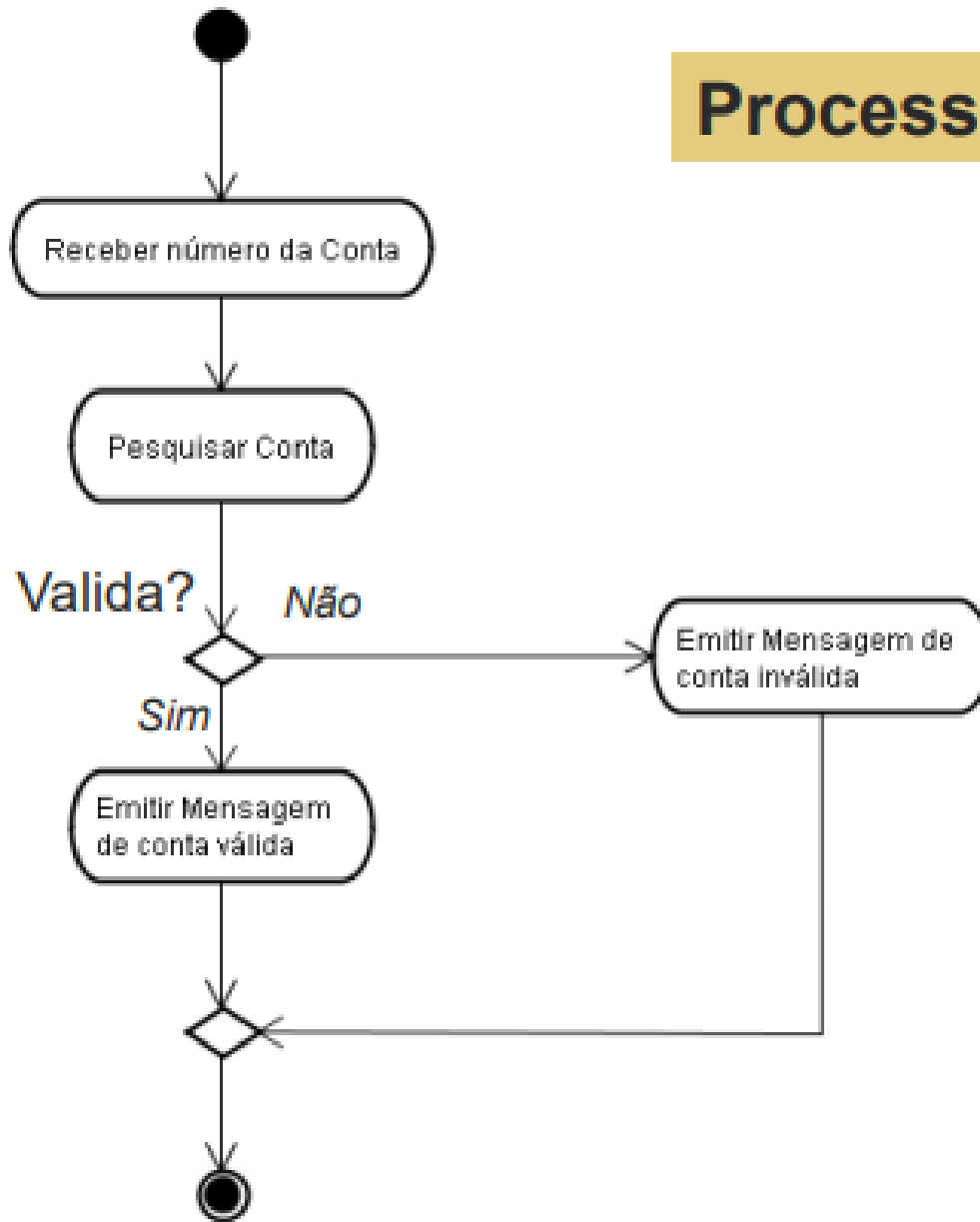
**Classe: Conta**  
**Processo: Encerrar Conta**

# Diagrama de Atividades

- Descreve as atividades a serem executadas para a conclusão de um processo
- Concentra-se na representação do fluxo de controle de um processo



## Processo: Validar Conta



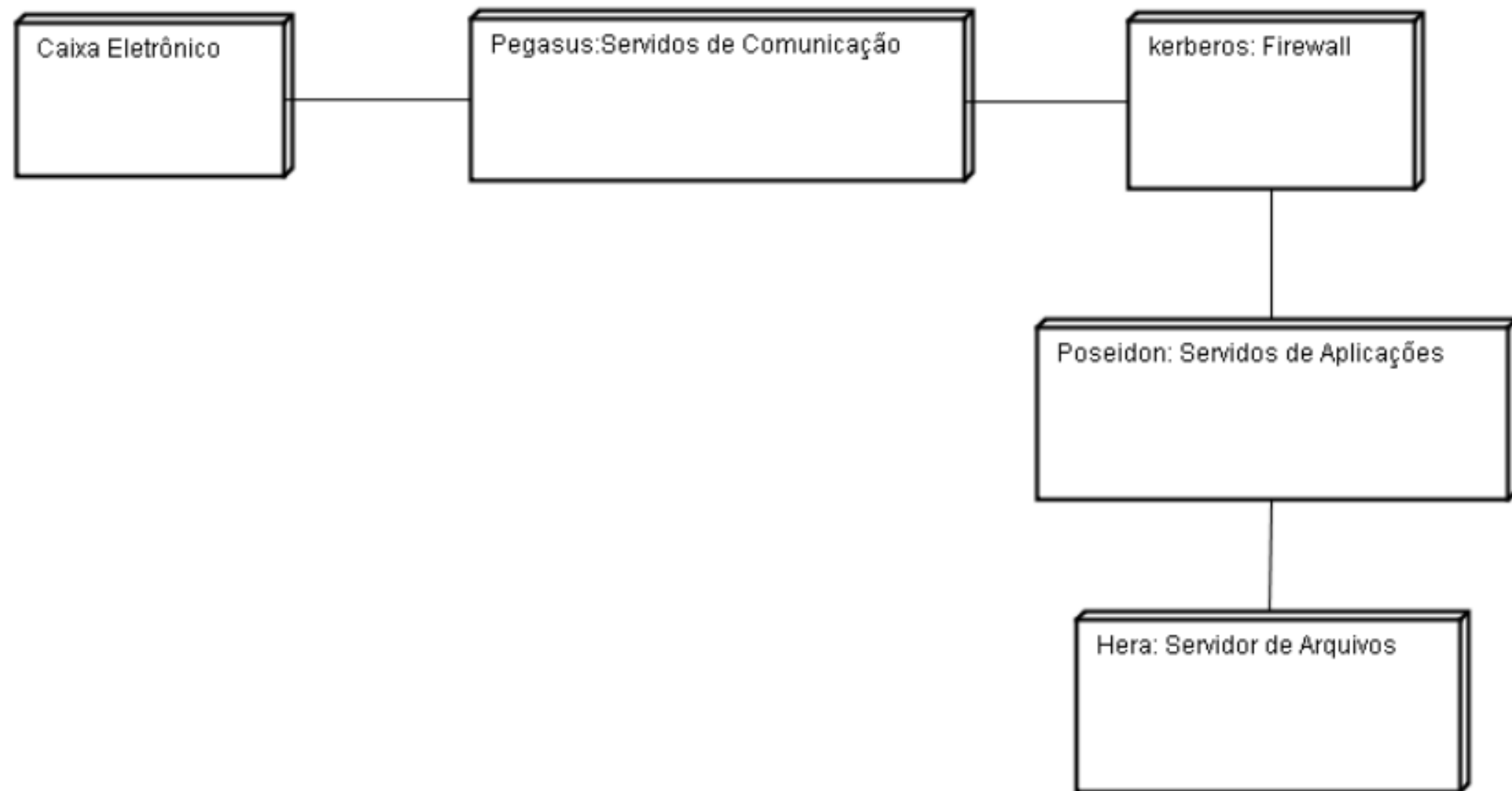
# Diagrama de Componentes

- Este Diagrama representa os componentes do sistema
  - Um componente é uma patê lógica e substituível do sistema
- Os componentes serão implementados como
  - Classe de código\_fonte
  - Bibliotecas
  - Arquivos de ajuda, etc



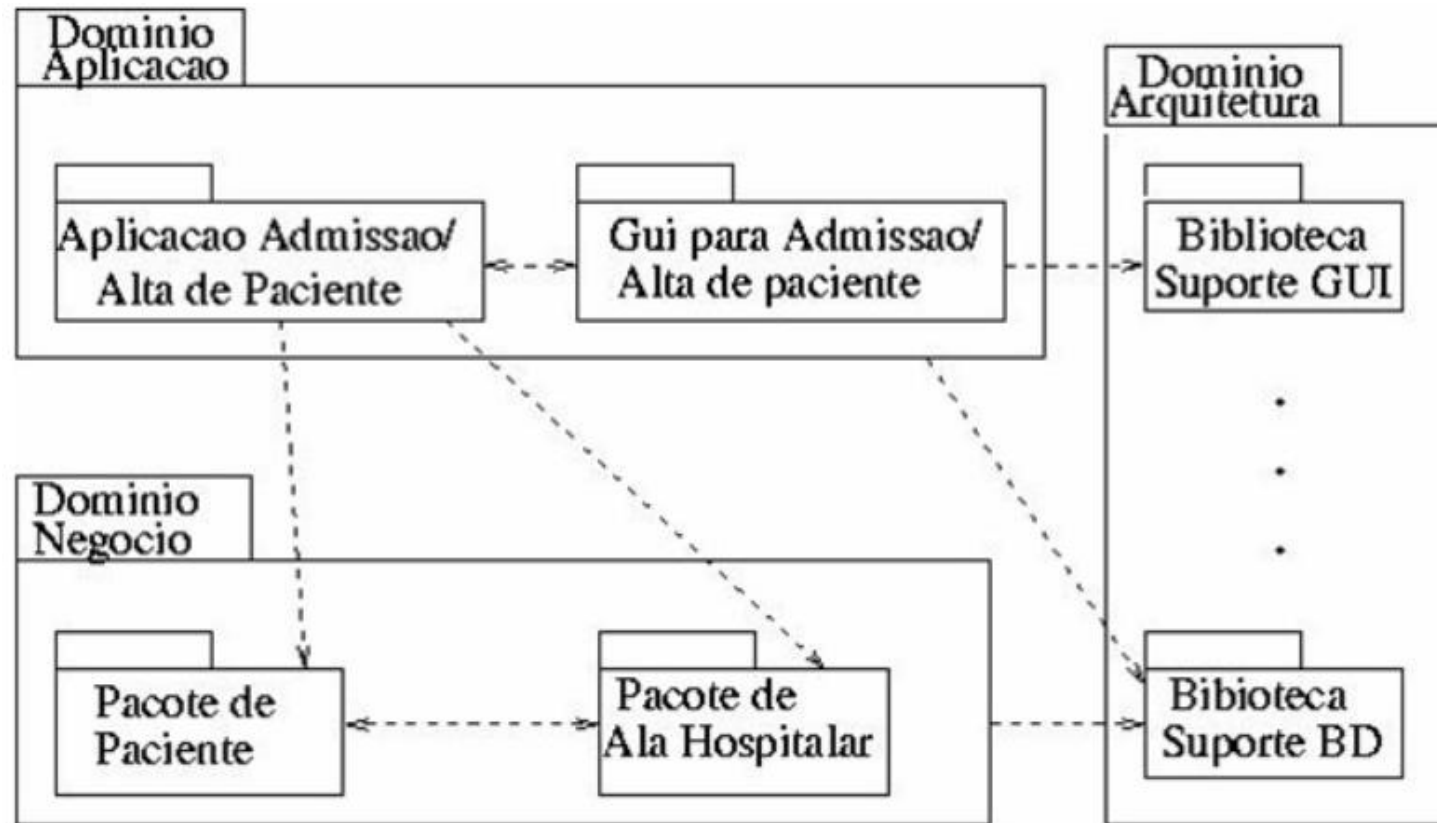
# Diagrama de Implantação

- Determina as necessidades de hardware
- Características físicas do sistema
  - Servidores
  - Estações
  - Topologias de Redes
  - Protocolos de comunicação, etc.



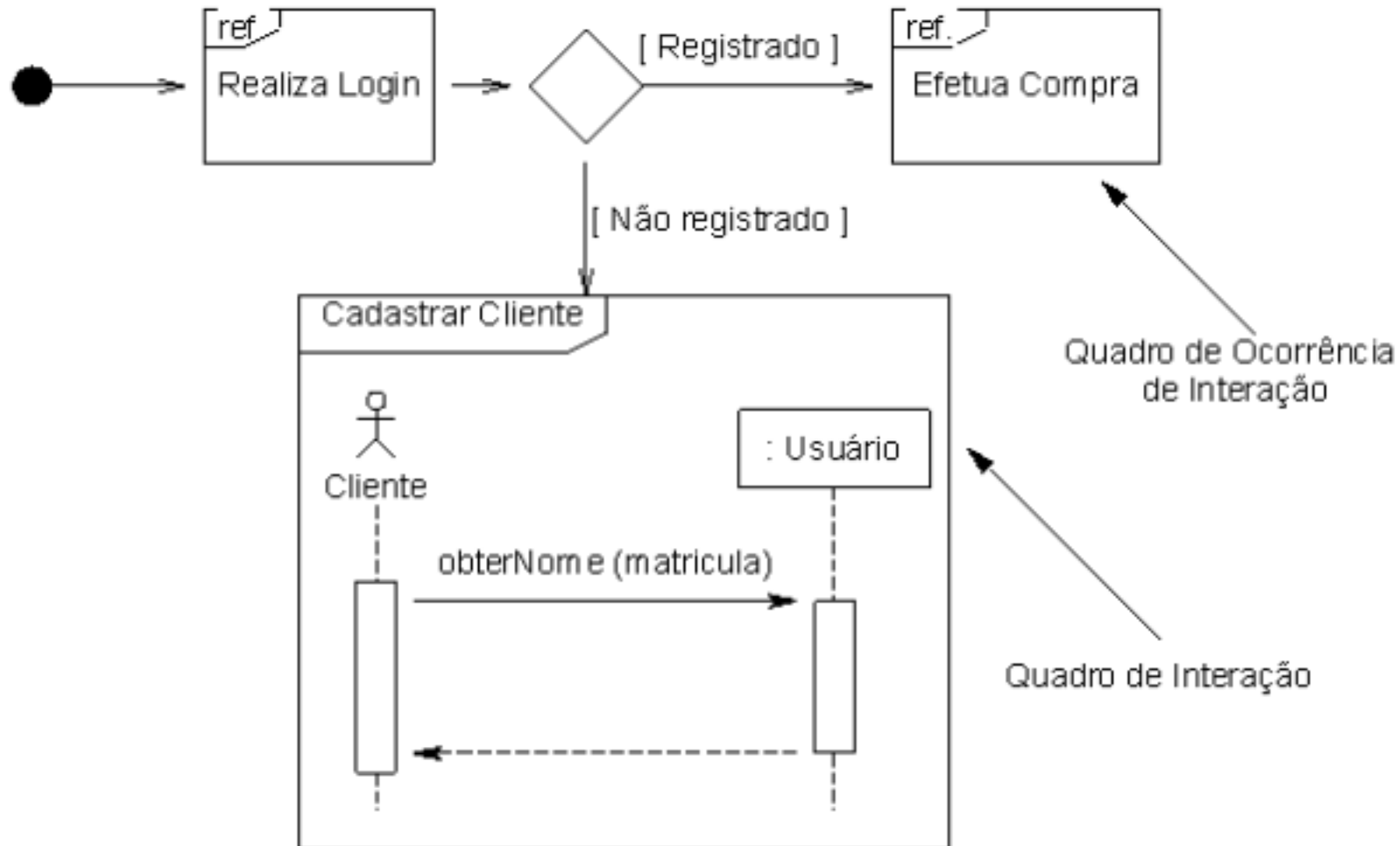
# Diagrama de Pacotes

- Um pacote é um conjunto de elementos agrupados. Esses elementos podem ser classes, diagramas, ou até mesmos outros pacotes.



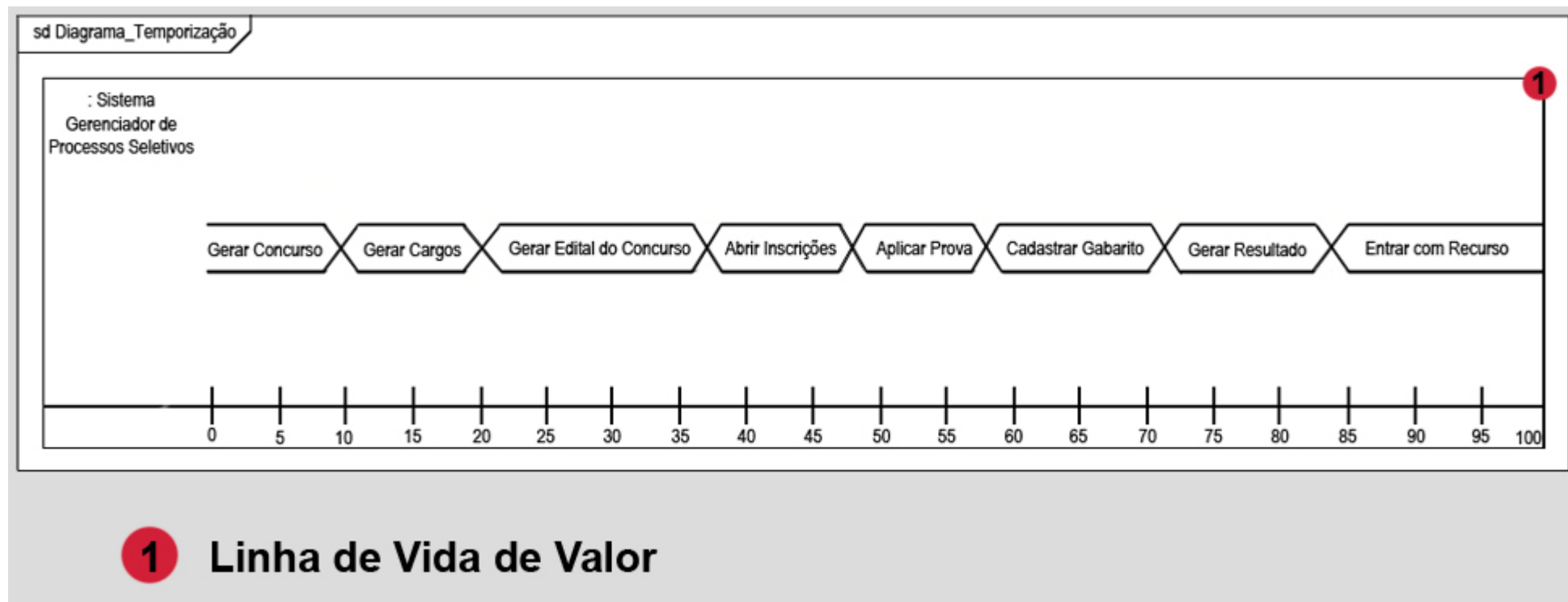
# Diagrama de Interação Geral

- Fornece uma visão geral dentro de um sistema ou processo de negócio.



# Diagrama de Temporização

- Descreve a mudança no estado ou na condição de uma instância de uma classe ou seu papel durante um período de tempo.







# Engenharia de Software I