

# Desenho Conceitos e Princípios



Aula 12

# Sumário

- # O quê é?
- # Propósitos do desenho
- # Desenho
  - ▣ De componentes
  - ▣ Da interface
  - ▣ Desenho Arquitectónico
  - ▣ Desenho de dados
- # Conceitos do desenho
  - ▣ Abstracção, refinação, modularidade, hierarquia de controlo, divisão estrutural, estrutura de dados, procedimento de software, encapsulamento
- # Modelo de desenho
- # Documentação do desenho



# O quê é?

---

# Representação significativa de engenharia de algo que vai ser construído

# Propósitos

---

- # Plataforma tecnológica
- # Base da programação
- # Decomposição da programação
- # Capturar interfaces
- # Facilitar a visualização e o raciocínio
- # Abstracção do sistema

# Desenho de dados

- # Transforma os diagramas E-A ou os diagramas de classe em:
  - ▣ estruturas de dados a nível dos componentes de software
  - ▣ algoritmos necessários para a sua manipulação
  - ▣ Esquemas de bases de dados, querys
  - ▣ Armazéns de dados
    - ▣ transacções, *datawarehouses*, *data-mining*

# Desenho arquitectónico

- # Relação entre os elementos estruturais do software, **padrões de desenho** que podem ser utilizados
- # Estilos arquitectónicos
  - ▣ Arquitecturas baseadas nos dados
  - ▣ Arquitecturas baseadas nos fluxos de dados
  - ▣ Arquitecturas de chamada e retorno
    - ▣ Chamada principal/subprograma
    - ▣ RPC
    - ▣ Arquitecturas OO
    - ▣ Arquitecturas n-tier

# Desenho da interface do utilizador

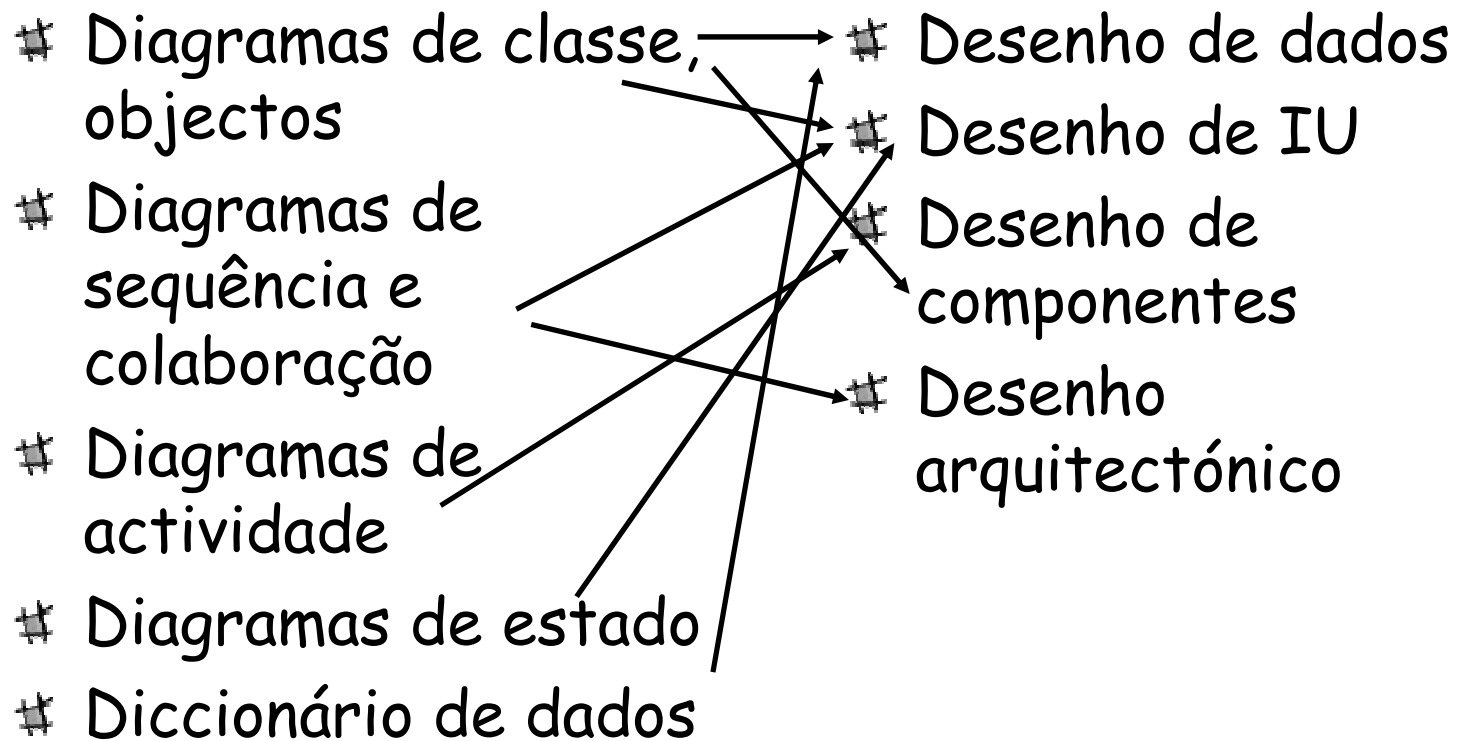
- # Desenho da interacção pessoa-máquina
- # Usabilidade: facilidade, comodidade de uso, aceitação do utilizador
- # Algumas normas de usabilidade
  - ▣ Estado do sistema visível
  - ▣ Minimizar carga cognitiva
  - ▣ Dar controlo ao utilizador
  - ▣ Permitir a detecção e recuperação de erros
  - ▣ Desenho minimalista

# Desenho de componentes

---

- # Componentes:
  - ▣ funções, subrotinas, objectos
- # Interfaces
- # Modularidade
- # Reutilização

# Da análise ao desenho..



# Conceitos do desenho

---

- # Abstracção
- # Refinamento
- # Modularidade
- # Arquitectura do software
- # Hierarquia de controlo
- # Divisão estrutural
- # Estrutura de dados
- # Procedimento de software
- # Ocultação de informação

# Desenho modular eficaz

---

- # Independência funcional:  
modularidade+abstracção+ocultação de  
informação
- # Critérios de medição
  - ▣ Coesão
  - ▣ Acoplamento
  - ▣ Desejável: alta coesão + baixo acoplamento

# Documentação do desenho

---

- # Especificações de desenho
  - ▣ Estruturas de dados
  - ▣ Interface do utilizador
  - ▣ Arquitectura do sistema
  - ▣ Especificação de componentes

# O modelo de desenho

---

- # Realização de use-cases através de:
  - ▣ Diagramas de classe, objectos, actividades
- # Diagramas de arquitectura
  - ▣ Diagramas de componentes
  - ▣ Diagramas de instalação

# Classes de desenho

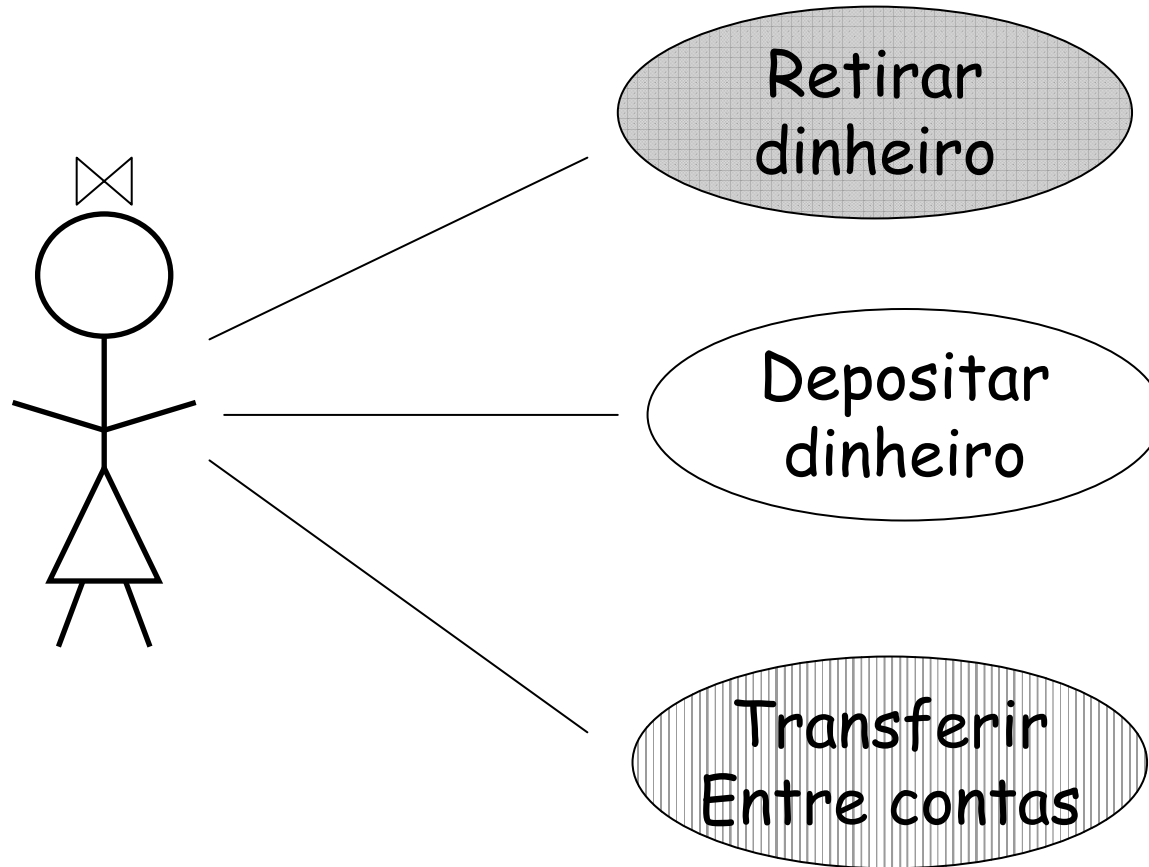
---

- # Especificada na linguagem de programação seleccionada
- # Visibilidade das propriedades
- # Relação directa com as classes implementadas

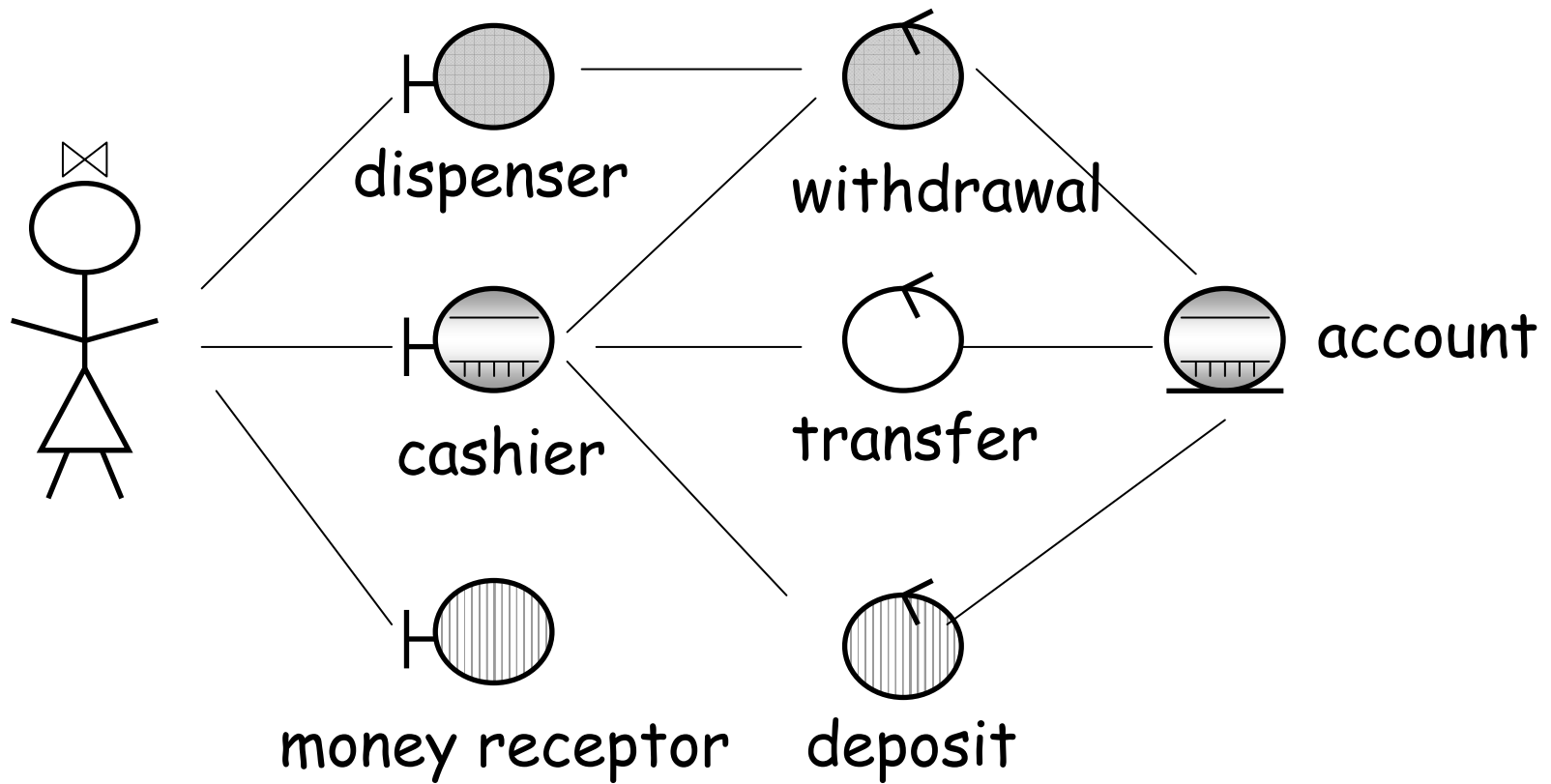
# Análise vs Desenho

- # Modelo conceptual
- # Independente da tecnologia
- # Três tipos de classes: entity, boundary e control
- # Menos formal
- # Mais económico
- # Menos camadas
- # Pouco dinâmico
- # Esquema do desenho
- # Não é mantido ao longo de todo o processo
- # Define apenas estrutura essencial
- # Modelo físico
- # Específico da tecnologia
- # Muitos tipos de classe
- # Mais formal
- # Mai caro
- # Mais camadas
- # Muito dinâmico
- # Desenho completo
- # Mantido ao longo do processo
- # Da forma ao sistema final, tenta preservar estrutura da análise

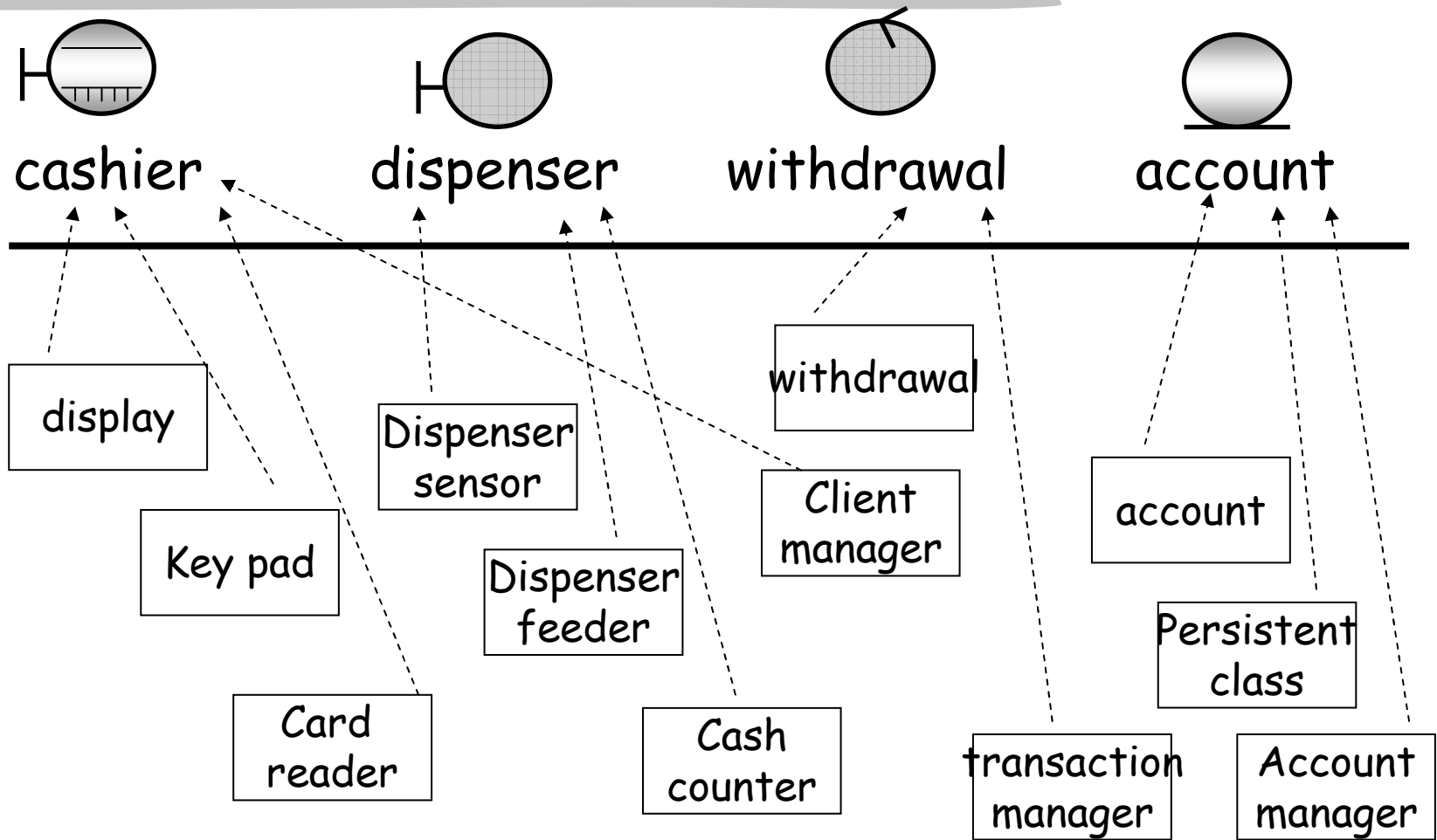
# Exemplo: multi-banco



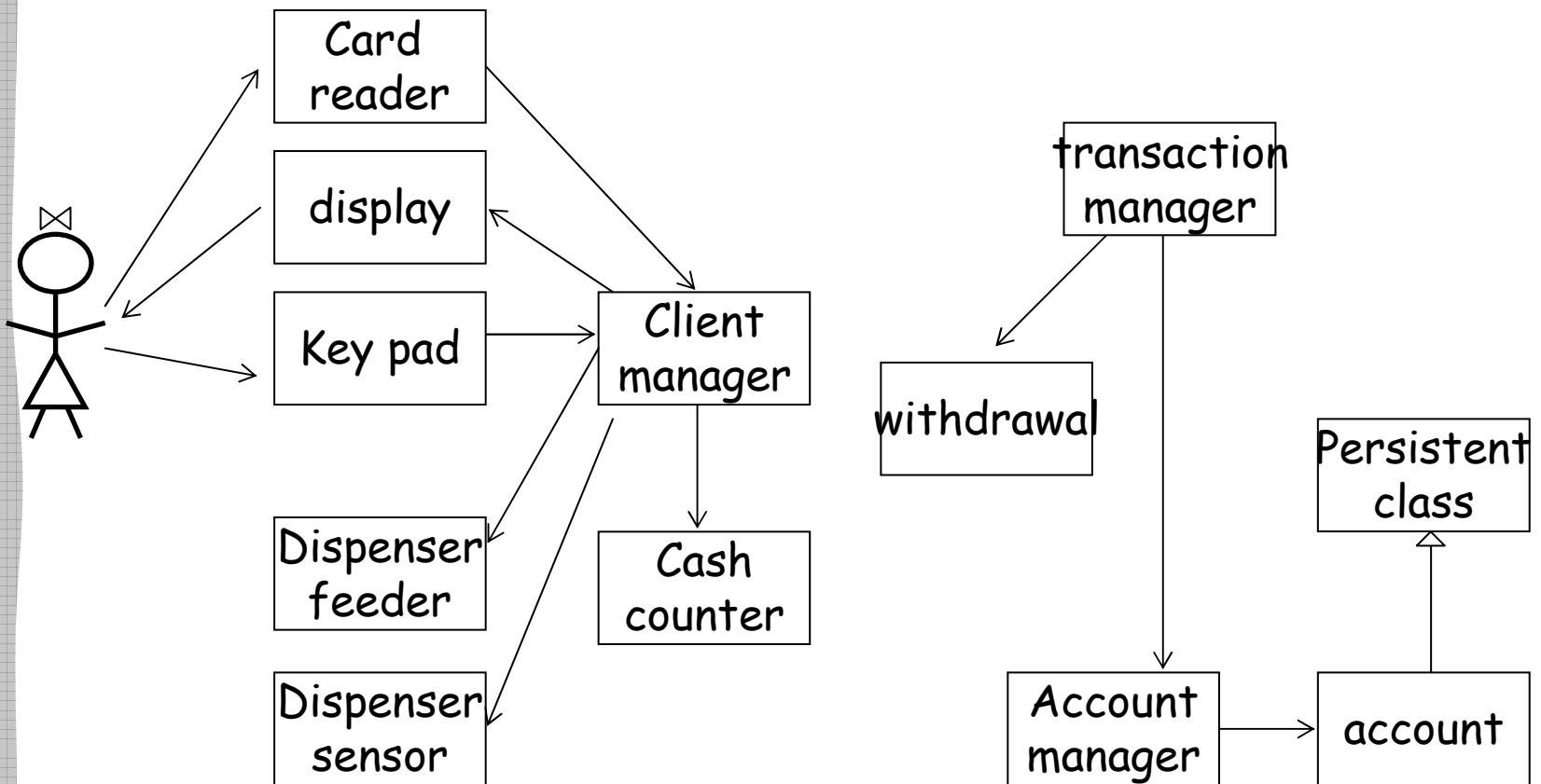
# Modelo de análise



# Modelo de desenho i



# Modelo de desenho ii



# Modelo de desenho iii

