UML

Fernando Pedrosa – fpedrosa@gmail.com

Bibliografia

- **Boch, Jacobson, Rumbaugh;** UML – Guia do Usuário; **Editora:** Elsevier; **Ano**: 2006
- **Martin Fowler;** UML Essencial; **Editora:** Bookman; **Ano**: 2004

Unified Modeling Language

Linguagem de Modelagem Unificada

- Linguagem
  - Usada para expressar e comunicar idéias
  - Não é uma metodologia!
- Modelagem
  - Descrever um sistema em um alto nível de abstração
- Unificada
  - UML se tornou o padrão mundial para modelagem de sistemas – www.omg.org

O que é a UML?

- Linguagem gráfica para especificar, visualizar, construir e documentar os artefatos de software
- Vantagens
  - Usa notação gráfica: mais clara que a linguagem natural (imprecisa) e código (muito detalhado)
  - Ajuda a obter uma visão geral do sistema
  - **Não** é dependente de tecnologia
  - Diminui a fragmentação, aumenta a padronização

Evolução

Industrialização

Padronização

Unificação

Fragmentação

Ano Versão

2011: UML 2.4

2010: UML 2.3

2009: UML 2.2

2003: UML 2.0

2001: UML 1.4

1999: UML 1.3

1997: UML 1.0, 1.1

1996: UML 0.9 & 0.91

1995: Unified Method 0.8

OOSE (Jacobson)

Outros métodos

Booch (OOAD)

OMT (Rumbaugh)

UML 2.2

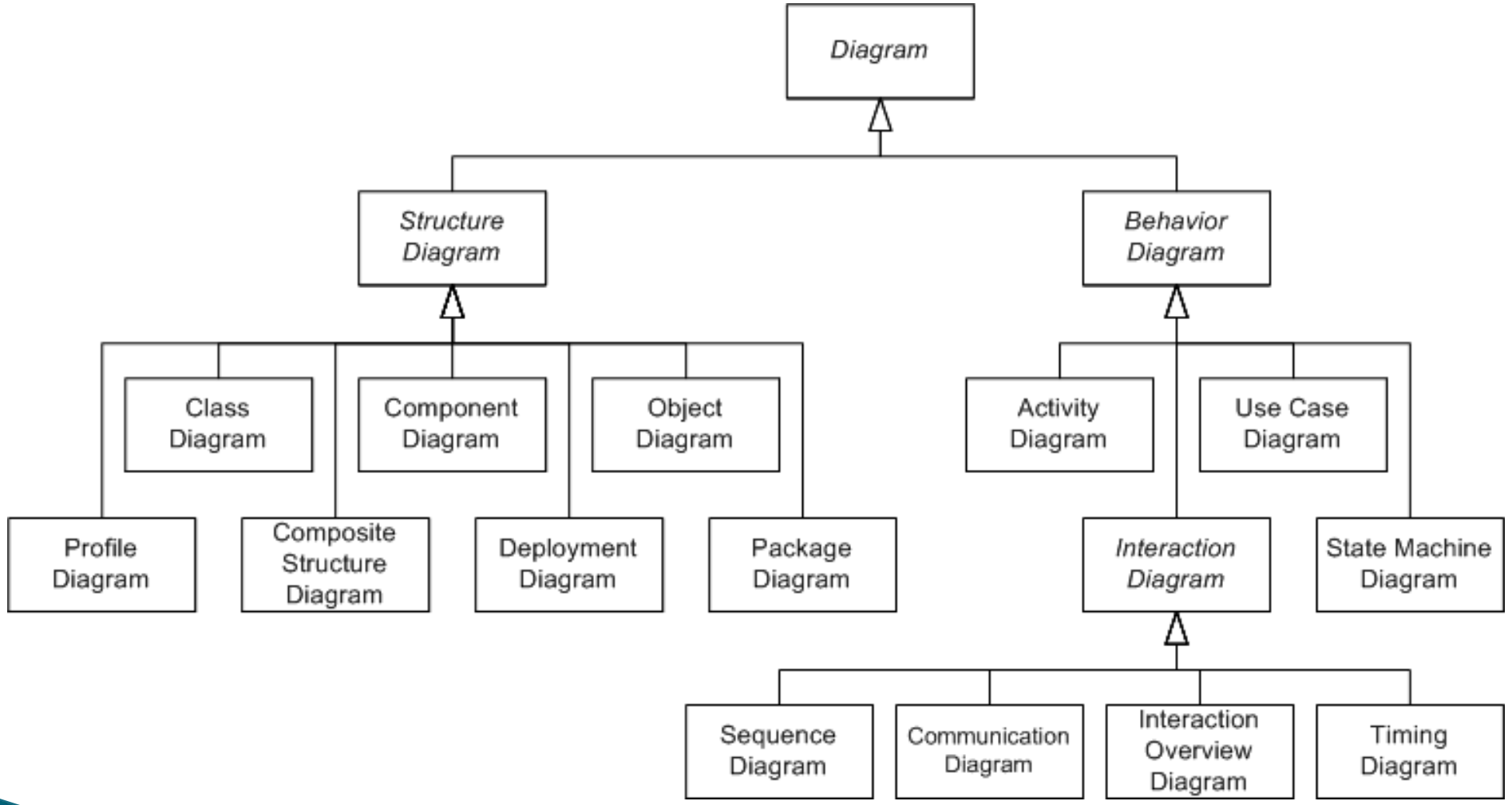


Diagram
Structure Diagram
Behavior Diagram
Class Diagram
Component Diagram
Object Diagram
Activity Diagram
Use Case Diagram
Profile Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Interaction Diagram
State Machine Diagram
Sequence Diagram
Communication Diagram
Interaction Overview Diagram
Timing Diagram

Tipos de Diagramas

- Diagramas estruturais
  - Mostram a estrutura estática do sistema e suas partes em diferentes níveis de abstração e como elas se relacionam
  - Não utilizam conceitos relacionados ao tempo
- Diagramas comportamentais
  - Mostram a natureza dinâmica dos objetos do sistema, que pode ser descrita como uma série de mudanças no sistema com o passar do tempo

Exercícios [1]

(Governo do ES – CESPE 2009)

[78] UML é um método para desenvolvimento de software que foi proposto para ser aplicado à análise e projeto de software orientados a objetos.

(EMBASA – CESPE 2009)

[94] Os diagramas em UML podem ser estáticos ou dinâmicos. O diagrama de classes é um exemplo de um diagrama dinâmico.

(SERPRO – CESPE 2008)

[101] UML (universal modelling language) é uma linguagem de modelagem proprietária que pode ser utilizada no desenvolvimento de sistemas de maneira intuitiva para visualização de objetos.

Exercícios [1]

(CGU – ESAF 2008)

[31] – A linguagem de Modelagem Unificada (UML) emergiu como notação de diagramação de padrão, de fato e de direito, para a modelagem orientada a objetos. Desta forma, a sentença que conceitua apropriadamente a UML, segundo o OMG–Object Management Group, é

a) um método para especificar e modelar os artefatos dos sistemas.

b) um processo de especificação e modelagem de sistemas orientados a objeto.

c) uma linguagem para implementar os conceitos da orientação a objetos.

d) uma linguagem visual para especificar, construir e documentar os artefatos dos sistemas.

e) um método comum para a representação da orientação a objetos.

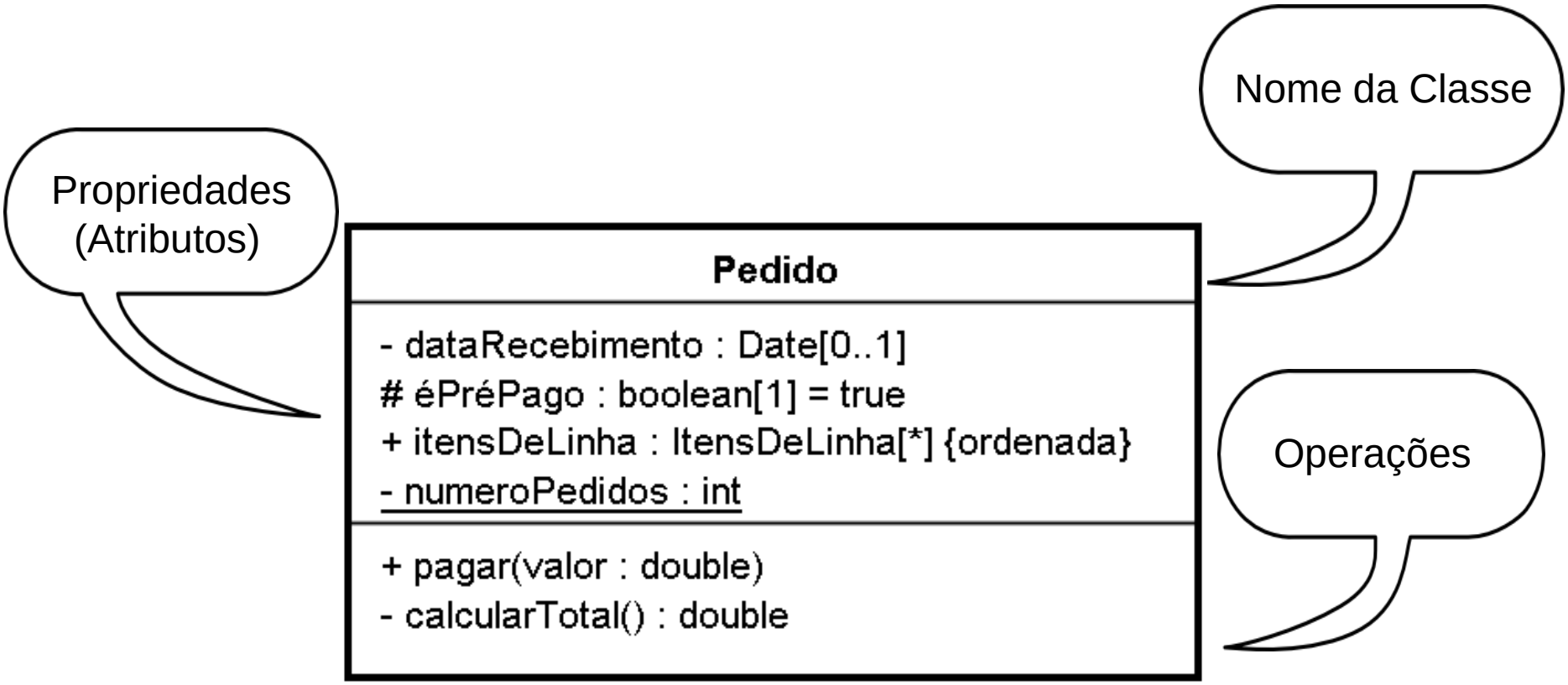
Diagramas Estruturais (estáticos)

- **Diagrama de Classes**
- Diagrama de Objetos
- Diagrama de Componentes
- Diagrama de Pacotes
- Diagrama de Implantação
- Diagrama de Estrutura Composta
- Diagrama de Perfis (UML 2.2)

Diagrama de Classes

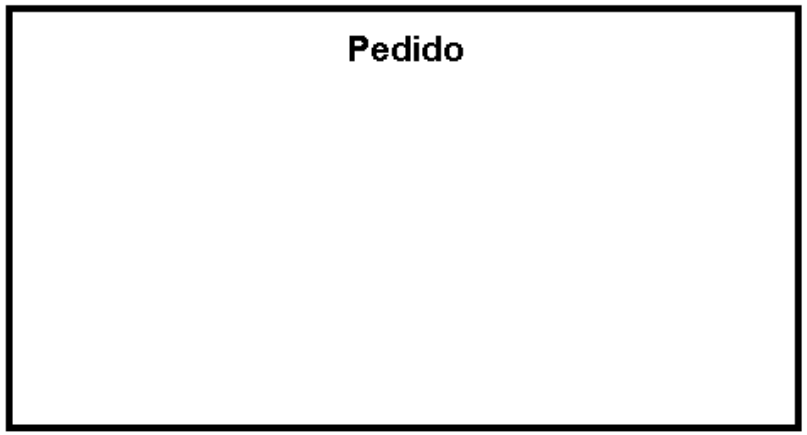
- É um diagrama estático da UML que reúne os elementos mais importantes de um sistema orientado a objetos
- Exibe um conjunto de classes, interfaces e seus relacionamentos
- As classes especificam tanto as propriedades quanto os comportamentos dos objetos

Estrutura da Classe

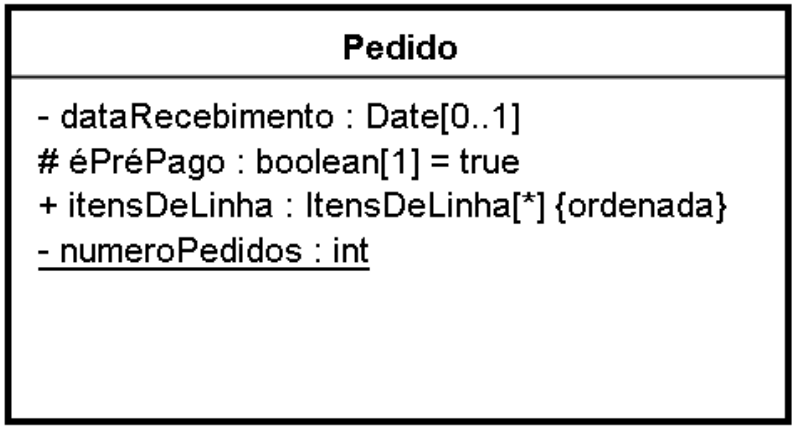


Três formas de representação

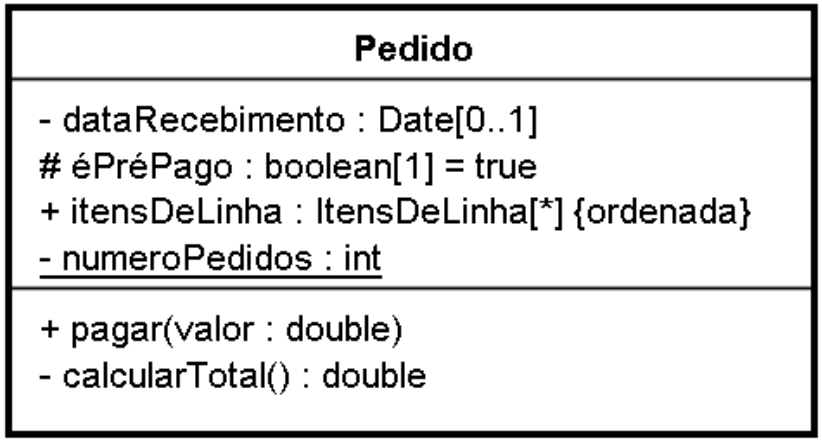
Nome

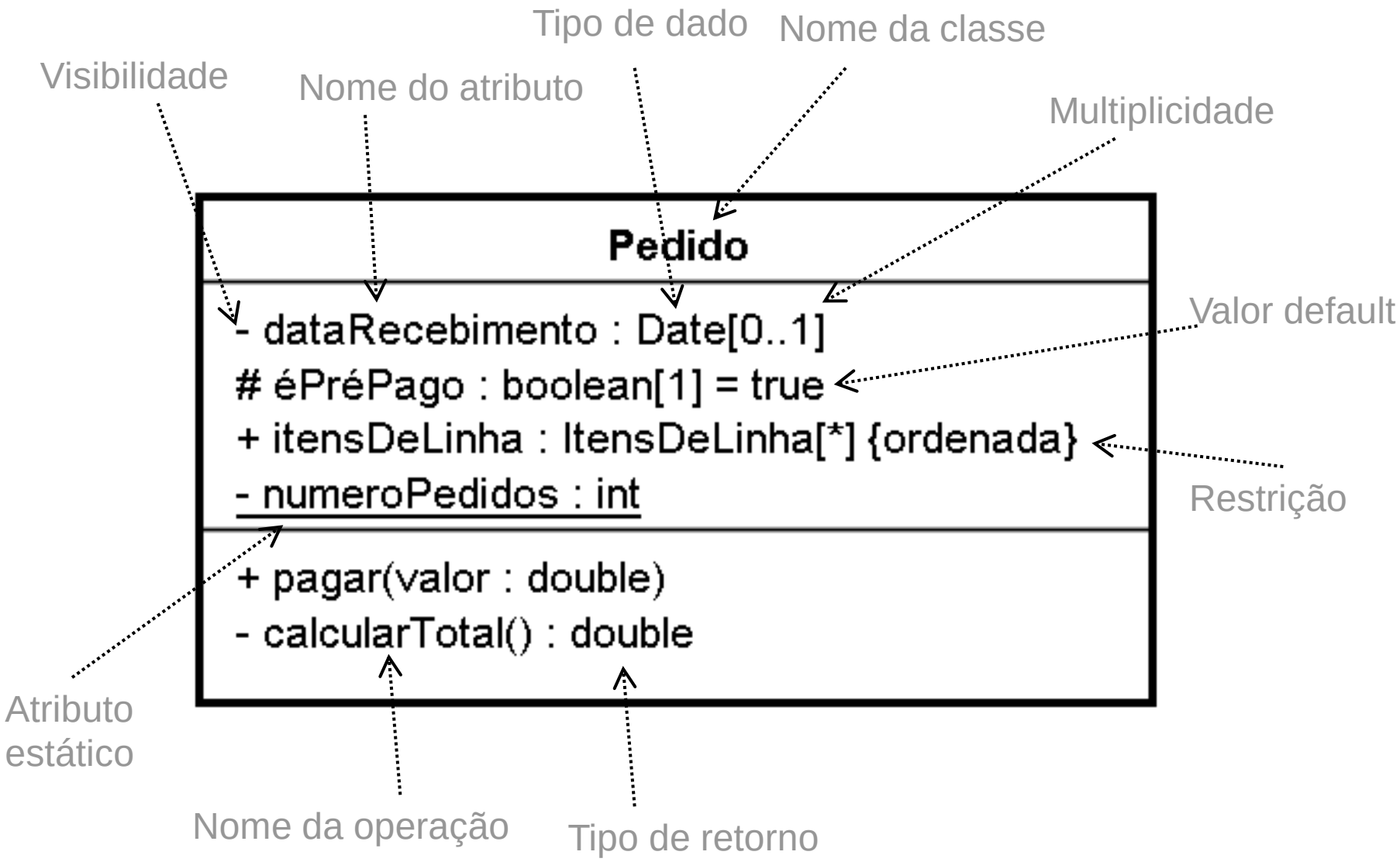


Nome + Atributos



Nome + Atributos + Operações




Tipo de dado
Nome da classe
Visibilidade
Nome do atributo
Multiplicidade
Pedido
- dataRecebimento : Date[0..1]
ésPréPago : boolean[1] = true
+ itensDeLinha : ItensDeLinha[*] {ordenada}
- numeroPedidos : int
+ pagar(valor : double)
- calcularTotal() : double
Valor default
Restrição
Atributo estático
Nome da operação
Tipo de retorno

Atributos

Notação completa:

Visibilidade nome: tipo [multiplicidade] = valor_default {lista de restrições}

Nome: corresponde ao nome do atributo
Tipo: domínio do atributo
Multiplicidade: indicação de quantos objetos podem preencher a propriedade [min..max]
Valor Default: valor do atributo, caso ele seja omitido no momento da criação
Restrição: permite indicar propriedades adicionais. {readOnly}, {ordered}, {unique}, etc.

Atributos – Escopo

- As propriedades (atributos) podem ter dois tipos de escopo
  - **Escopo de instância**: cada objeto tem o seu próprio valor para o atributo. É o escopo *default* da UML.
  - **Escopo de classe (estático)**: o valor do atributo é comum a todos os objetos daquela classe. Para denotar este escopo o atributo deve ser sublinhado.

Operações

Notação completa:

Visibilidade nome (lista de parâmetros): tipo-de-retorno {lista restrições}

Nome: corresponde ao nome da operação
Lista de parâmetros: são os parâmetros da operação.
Tipo de retorno: o tipo de dado retornado pela operação
Restrição: permite indicar propriedades adicionais.
ex: {query}.

Operações abstratas e estáticas

- Operações abstratas, ou seja, que não têm uma implementação específica, devem ser escritas em *itálico*
- Operações estáticas devem ser escritas com fonte sublinhada

Modificadores de Visibilidade

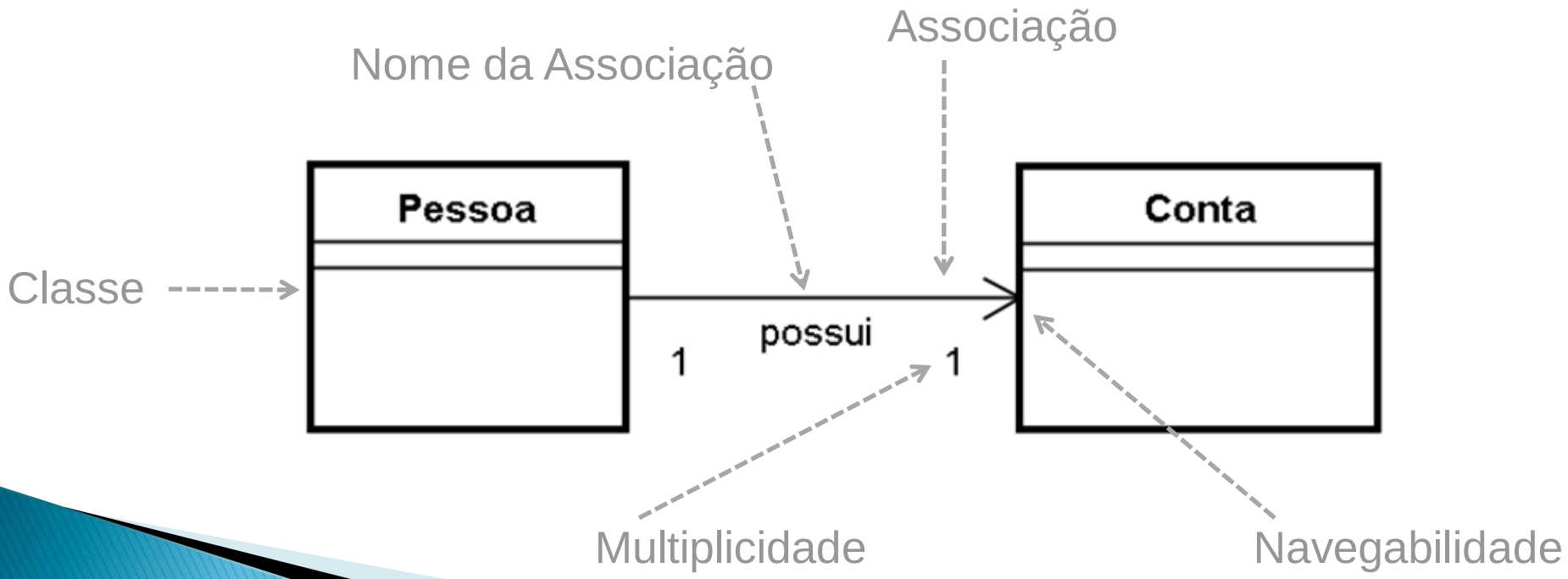
- Público (+)
  - O elemento é visível por qualquer classe
- Protegido (#)
  - O elemento é visível na própria classe e pelas subclasses da classe
- Pacote (~)
  - O elemento é visível apenas pela própria classe ou dentro do pacote onde a classe está localizada
- Privado (-)
  - O elemento é visível apenas pela própria classe

Relacionamentos

- Relacionamentos ligam classes entre si, criando relações lógicas
- Podem ser de:
  - **Associação**
    - Simples
    - Agregação
    - Composição
  - **Generalização**
  - **Dependência**
  - **Realização**

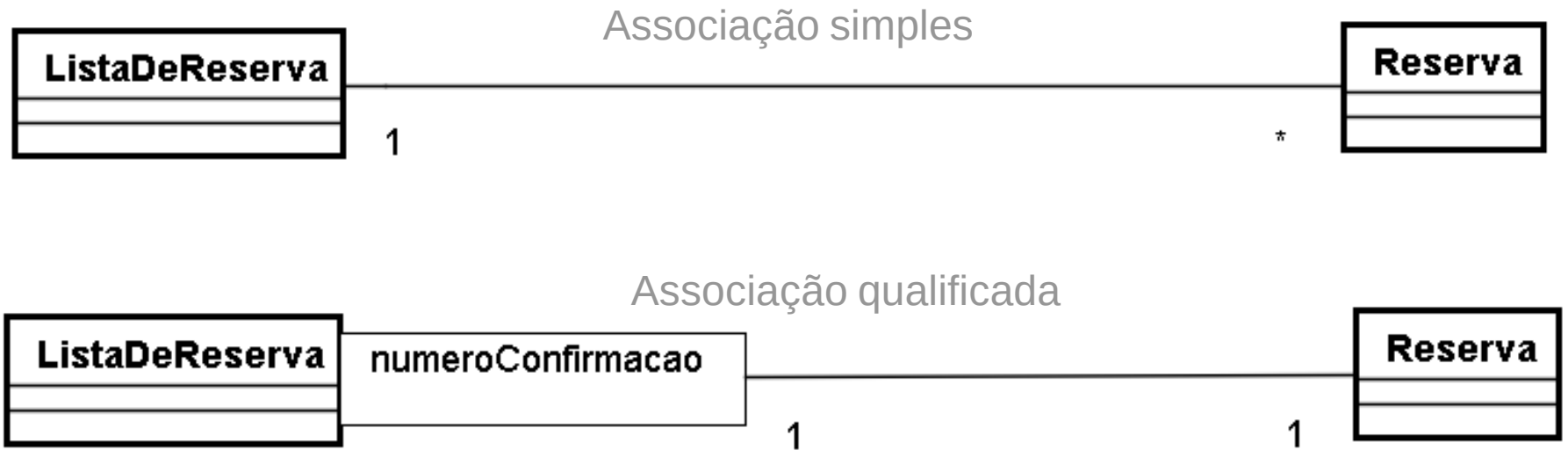
Associação simples

- Indica que objetos de um elemento estão ligados a objetos de outro elemento
- A navegabilidade pode ser unidirecional ou bidirecional



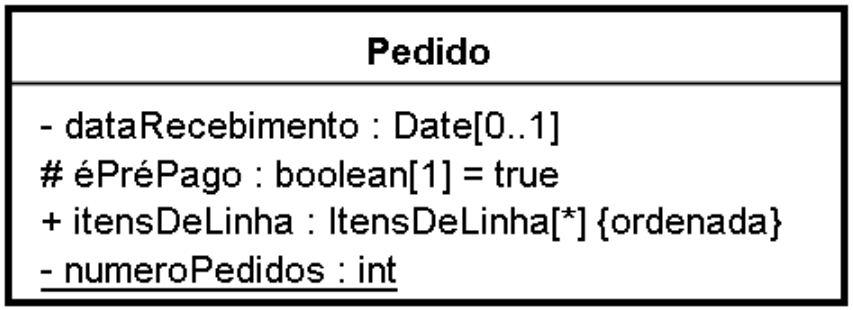
Associação qualificada

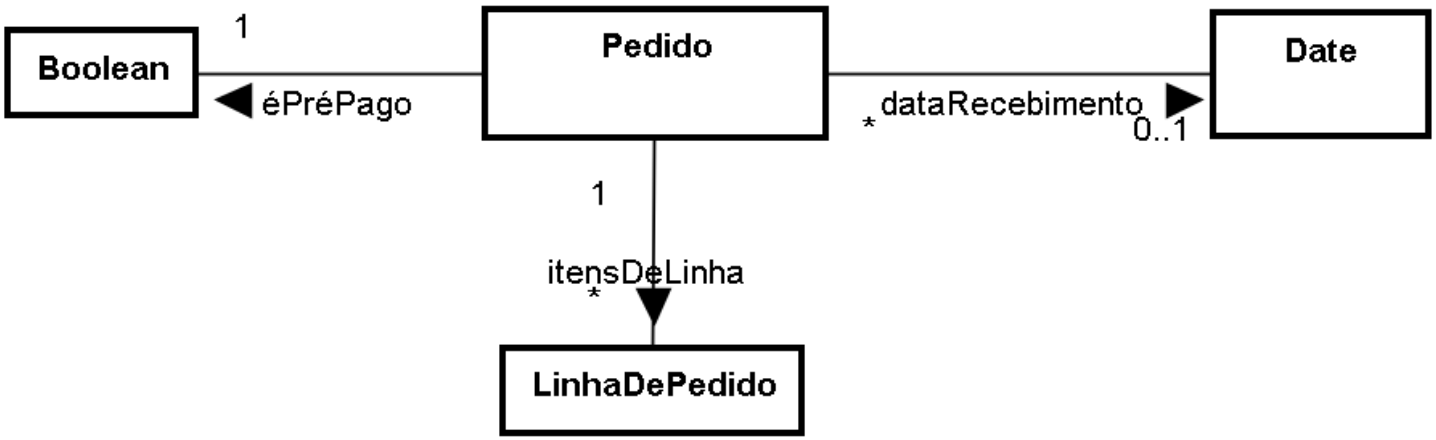
- Um qualificador de associação é um atributo do elemento-alvo capaz de identificar uma instância dentre as demais



Associação x Atributos

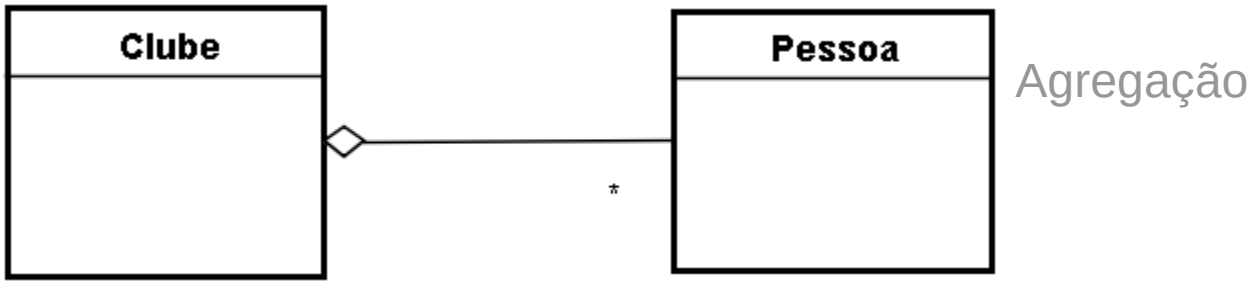
Uma associação pode mostrar as mesmas informações que um atributo





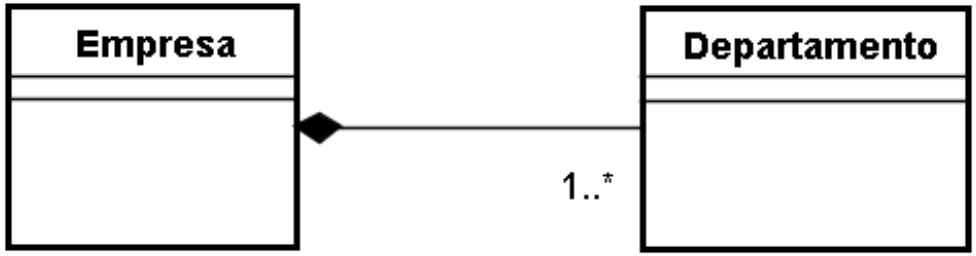
Agregação e Composição

- Relacionamentos “todo-parte”
- **Agregação**: a parte existe sem o todo
- **Composição**: o todo controla o ciclo de vida da parte, e ela não pode ser compartilhada em outros relacionamentos



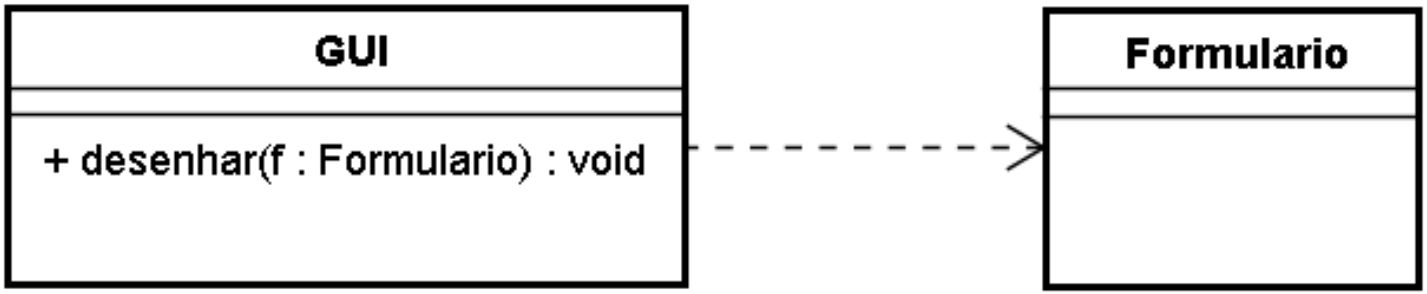
Agregação

Composição



Dependência

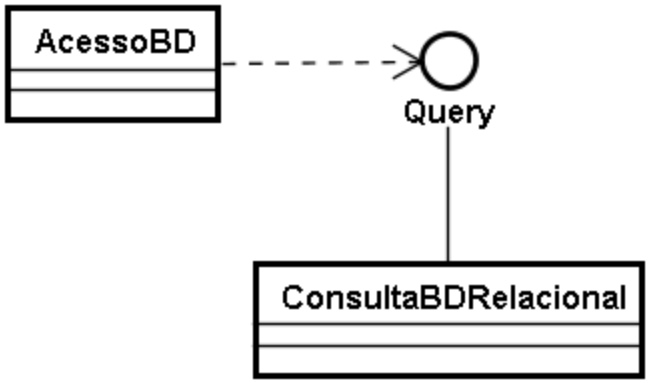
- Indica que mudança em um elemento pode causar mudanças no outro (uso)




```
public class GUI {

        public void desenhar(Formulario f) {
                f.pintarBotao();
                f.pintarMenu();
                f.pintarJanelas();
                ...
        }
}
```

Dependência

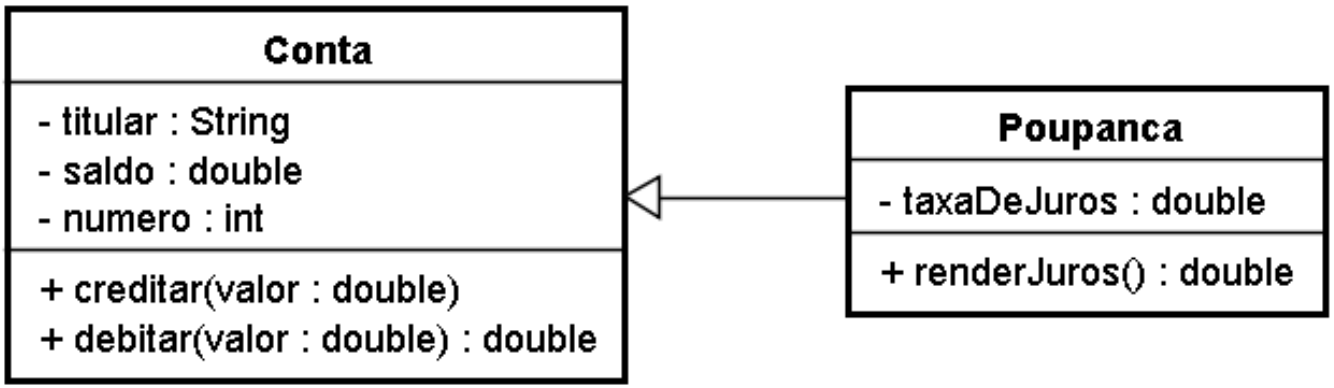
- Pode ocorrer, também, entre uma classe e uma interface




```
public class AcessoBD {
        private Conexao conexao;

        public void consultar(Query query) {
                query.iniciarConsulta();
                ...
        }
}
```

Generalização

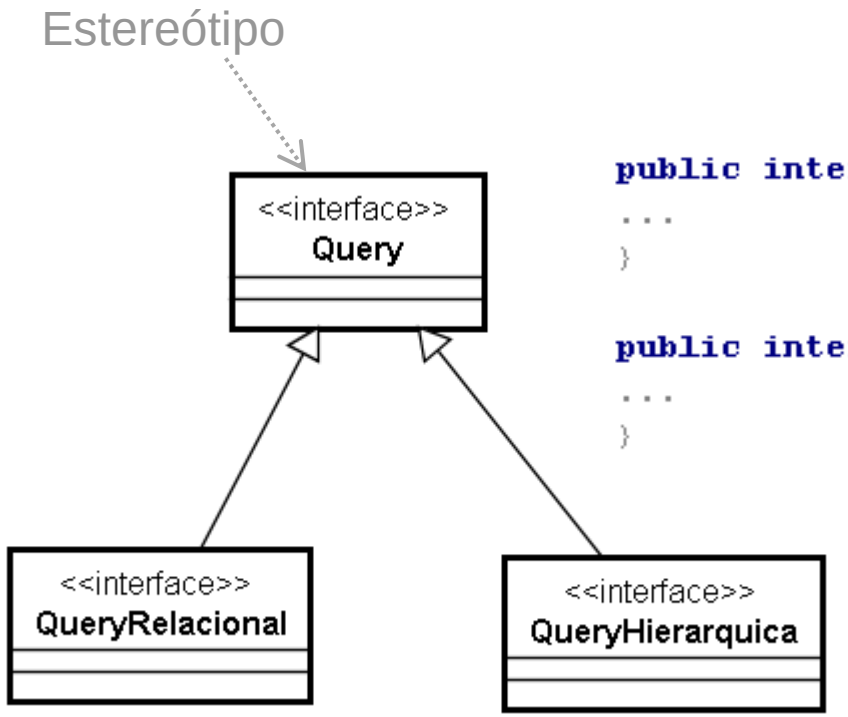
- Relacionamento “é um tipo de”




```
public class Conta {
        private String titular;
        private double saldo;
        private int numero;
        public void creditar(double valor) {...}
        public double debitar (double valor) {...}
}

public class Poupanca extends Conta {
        private double taxaDeJuros;
        public double renderJuros () {...}
}
```

Generalização

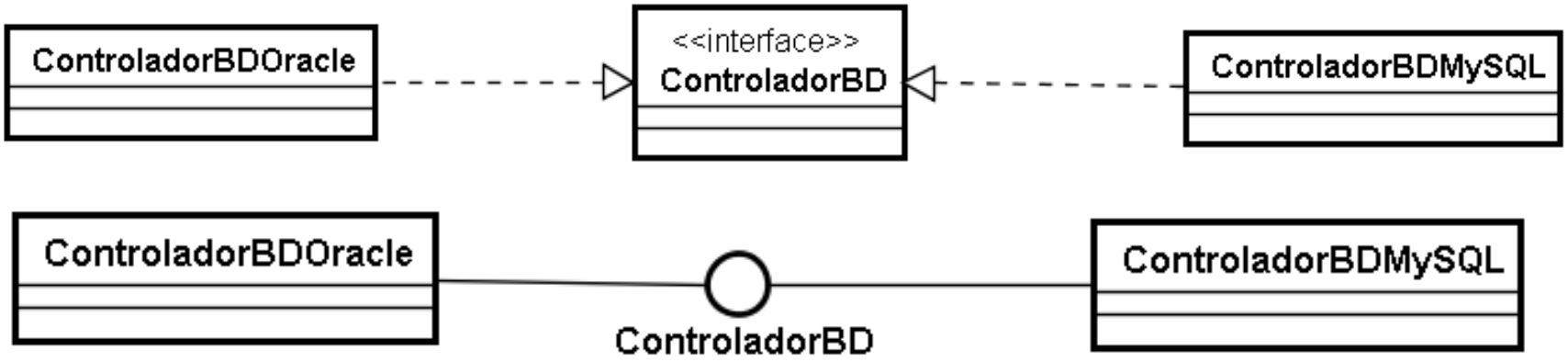
- Pode ocorrer, também, entre interfaces




```
public interface QueryRelacional extends Query {
...
}

public interface QueryHierarquica extends Query {
...
}
```

Realização (Interfaces)

- Há várias notações para realizações

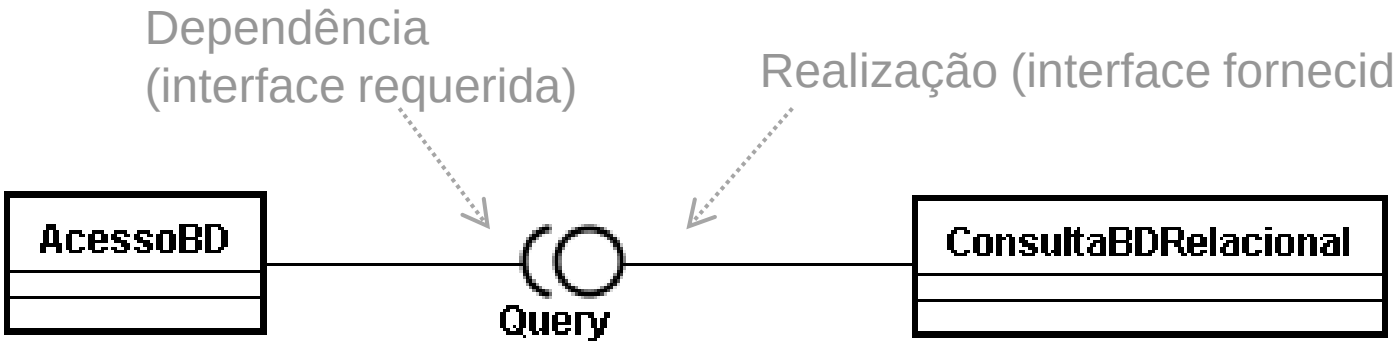



```
public interface ControladorBD {

        public void insert(String SQL);
        public Object select(String SQL);
        ...
}
```

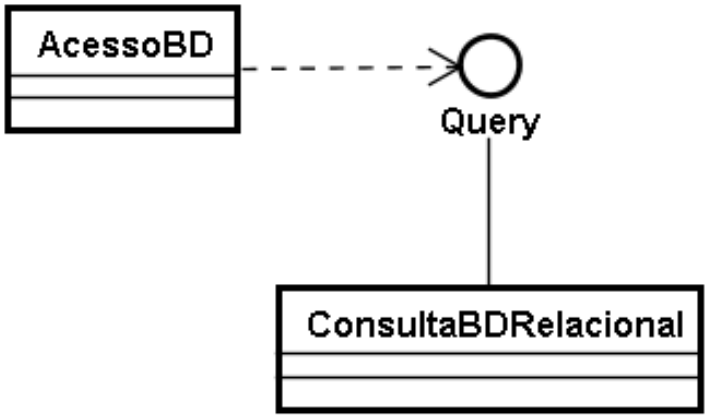
```
public class ControladorOracle
        implements ControladorBD {

        public void insert(String SQL) {
                ...
        }
        public Object select(String SQL) {
                ...
        }
}
```

Realização (Interfaces)

- A notação "bola-soquete" (UML 2.0) é utilizada para modelar uma dependência e uma realização entre duas classes e uma interface



Duas representações equivalentes entre si (semanticamente iguais)



Exercícios [2]

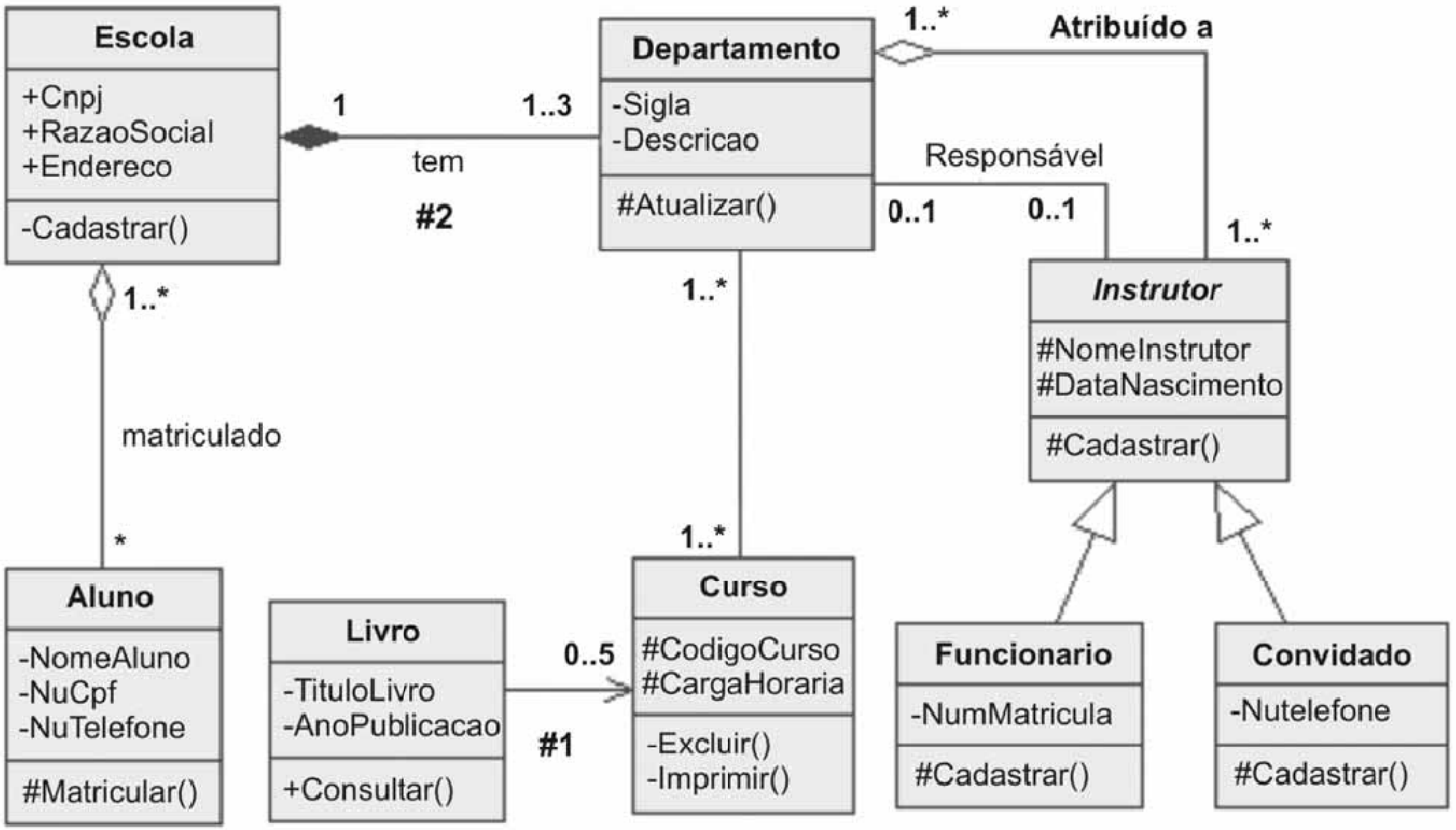
(MPE/AM – CESPE 2008)

[88] Na UML, uma agregação é um relacionamento que estabelece que uma classe define objetos que são parte de um objeto definido por outra classe.

[89] Em um diagrama de classes UML, uma associação entre classes pode ser documentada em termos da multiplicidade da associação.

Exercícios [2]

(TCU – CESPE 2009)



Exercícios [2]

[108] O método #Cadastrar() da classe Instrutor tem visibilidade do modo protegido tal que somente a classe possuidora Instrutor pode utilizá-lo.

[109] Na associação do tipo agregação composta identificado por #2, uma instância da classe Departamento pertence exclusivamente a uma única instância composta em Escola, e um objeto da classe Escola pode relacionar-se com, no máximo, três objetos da classe Departamento.

[110] Instrutor é uma superclasse abstrata; assim, o método #Cadastrar() oferece uma assinatura, que, no entanto, está incompleta, devendo ser implementada pelos métodos de mesmo nome nas suas classes-filhas.

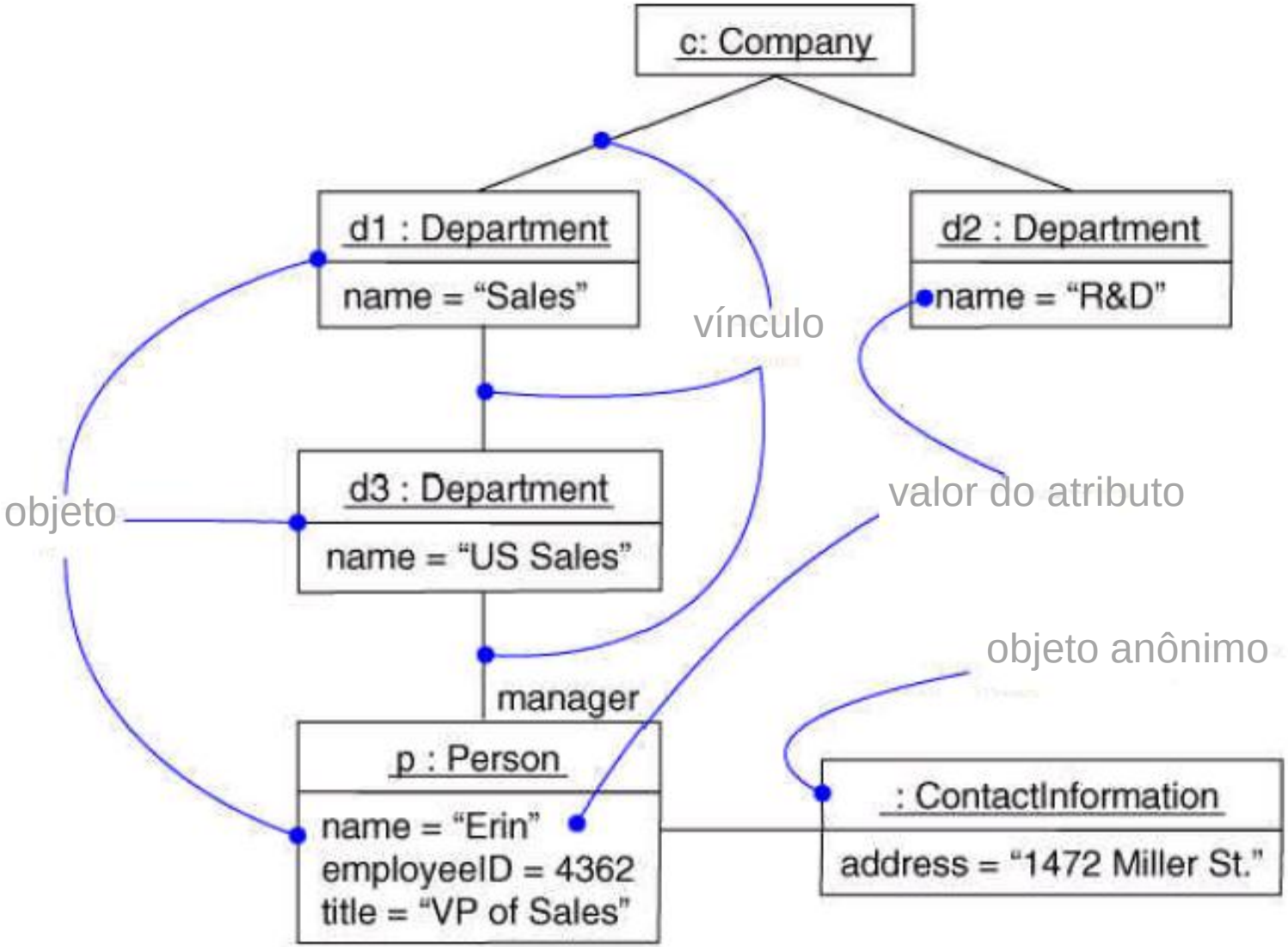
Diagramas Estruturais (estáticos)

- Diagrama de Classes
- **Diagrama de Objetos**
- Diagrama de Componentes
- Diagrama de Pacotes
- Diagrama de Implantação
- Diagrama de Estrutura Composta
- Diagrama de Perfis (UML 2.2)

Diagrama de Objetos

- O diagrama de objetos representa uma fotografia do sistema em um dado momento
- Mostra os vínculos entre os objetos conforme estes interagem e os valores dos seus atributos
- Pode ser visto como uma “instância” do diagrama de classe

Diagrama de Objetos



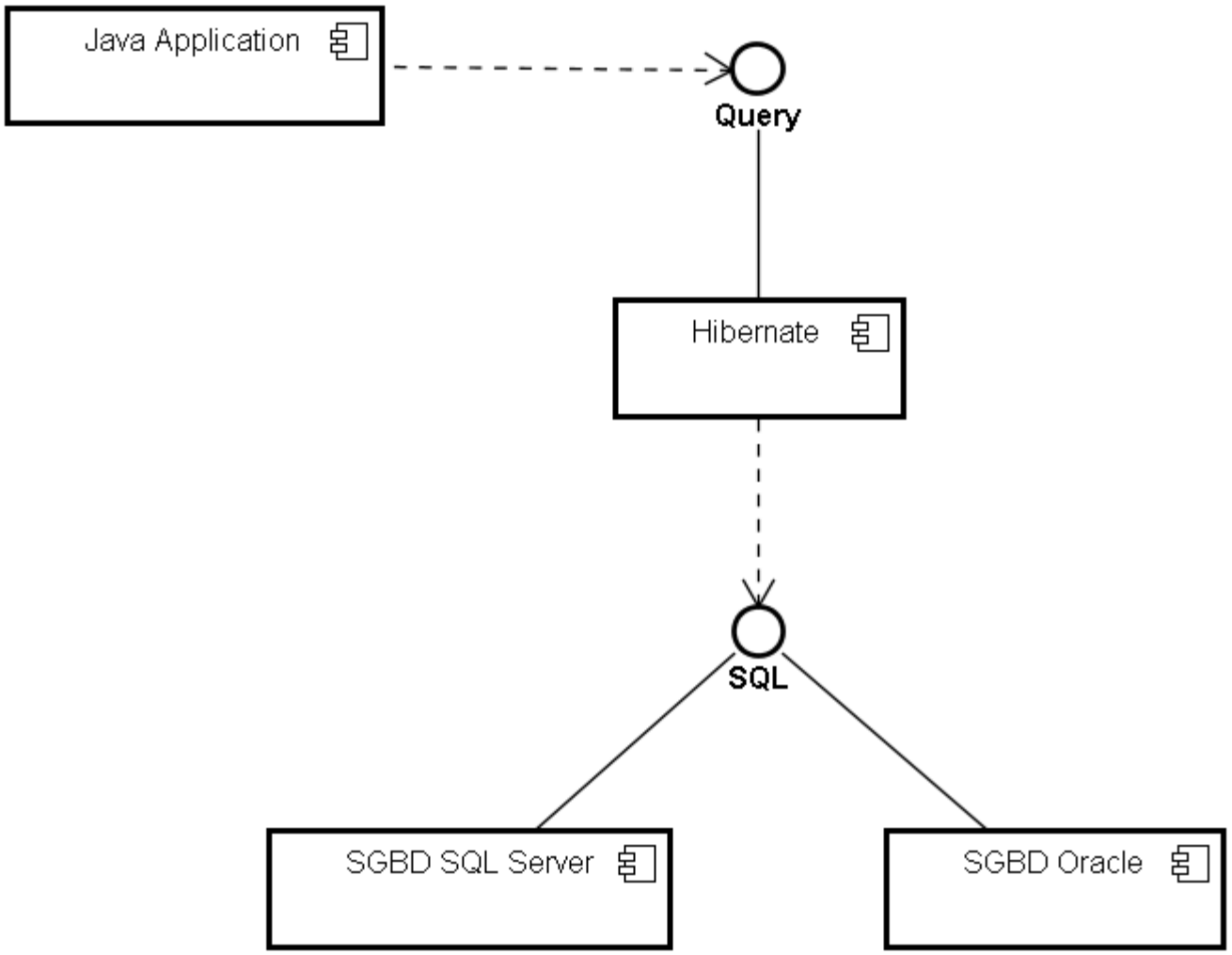
Diagramas Estruturais (estáticos)

- Diagrama de Classes
- Diagrama de Objetos
- **Diagrama de Componentes**
- Diagrama de Pacotes
- Diagrama de Implantação
- Diagrama de Estrutura Composta
- Diagrama de Perfis (UML 2.2)

Diagrama de Componentes

- Modela o sistema em termos de componentes e seus relacionamentos através de interfaces
- Decompõe o sistema em subsistemas que detalham a estrutura interna
- Alguns componentes existem em tempo de ligação, outros em tempo de execução

Diagrama de Componentes



Diagramas Estruturais (estáticos)

- Diagrama de Classes
- Diagrama de Objetos
- Diagrama de Componentes
- **Diagrama de Pacotes**
- Diagrama de Implantação
- Diagrama de Estrutura Composta
- Diagrama de Perfis (UML 2.2)

Diagrama de Pacotes

- Pacotes são estruturas que permitem agrupar qualquer construção da UML em estruturas de alto nível
- Pode mostrar:
  - Pacotes e suas dependências
  - Interfaces entre os pacotes
  - Generalizações entre pacotes

Diagrama de Pacotes

- Duas representações possíveis

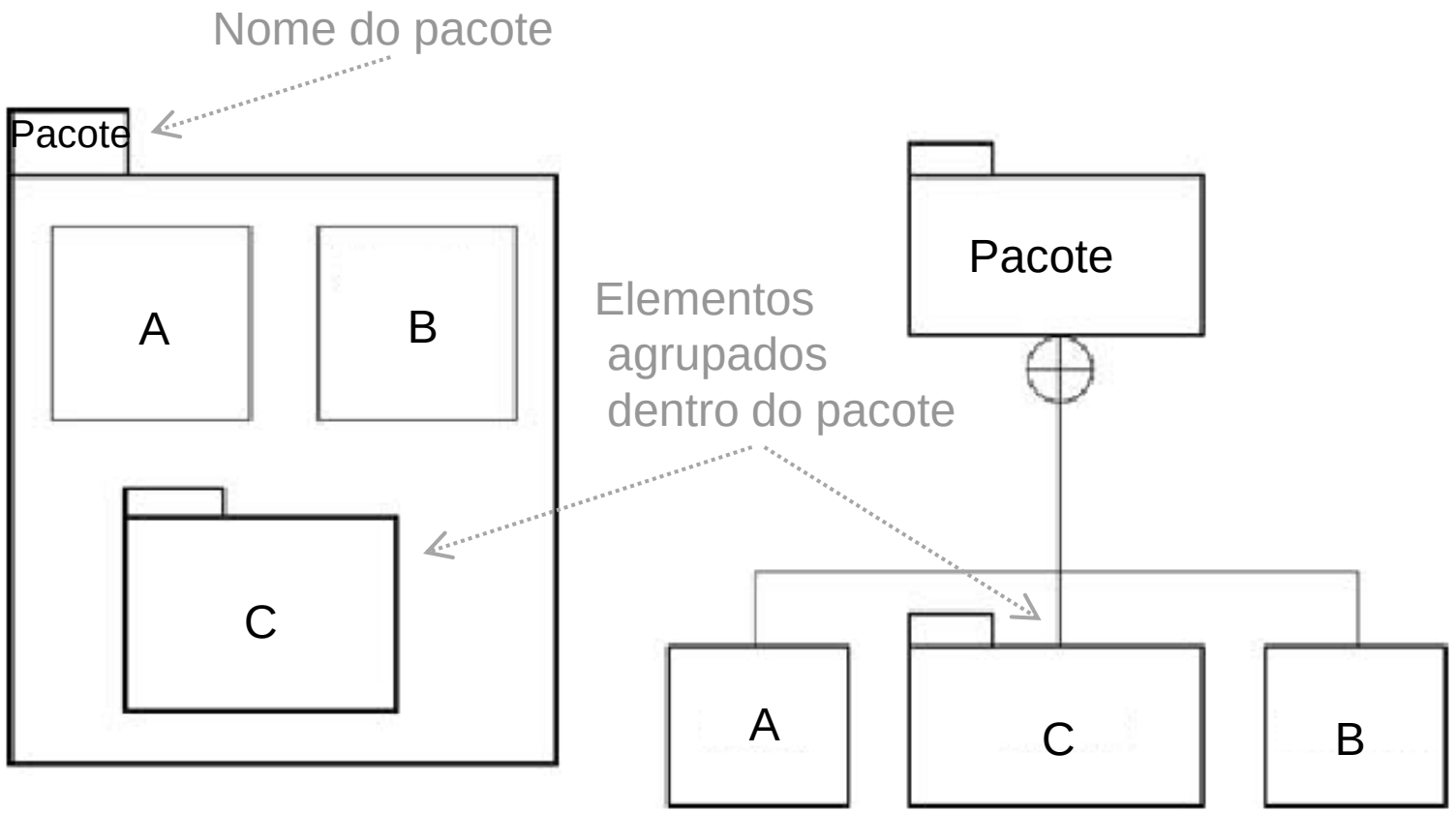
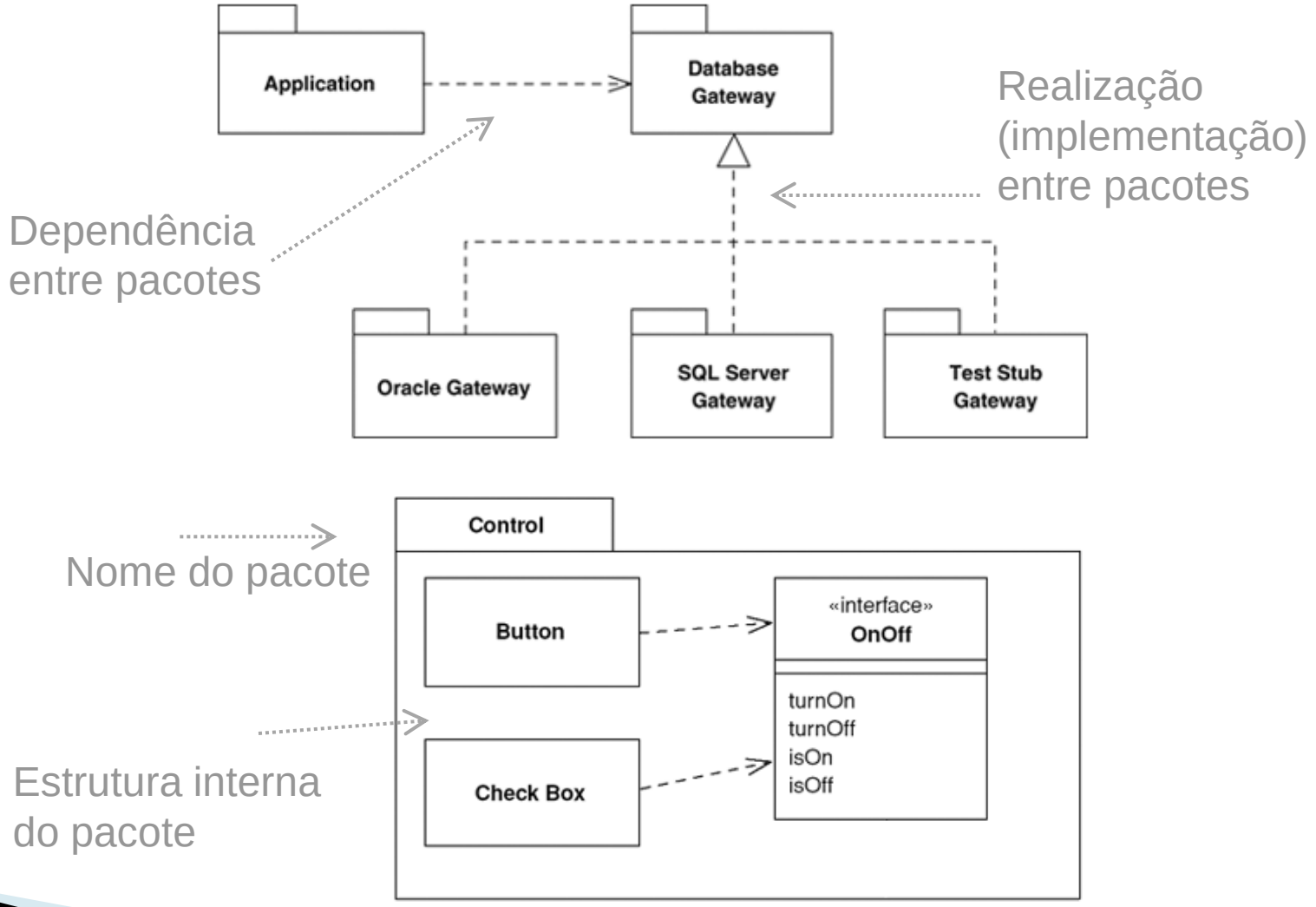


Diagrama de Pacotes



Exercícios [3]

(UNIPAMPA – CESPE 2010)

[106] O diagrama de objetos está amplamente associado ao diagrama de classes, sendo que o primeiro consiste em uma instância do segundo, em determinado momento da execução, ou seja, um diagrama de objetos descreve os objetos, os métodos, os atributos e seus valores, além dos vínculos entre os objetos, sendo ambos diagramas estruturais.

(TRE/TO – CESPE 2007)

[40] Um diagrama de componentes permite mostrar componentes de um sistema e as dependências entre eles. As dependências entre os componentes podem ser, por exemplo, dependências de compilação ou de comunicação.

Exercícios [3]

(EMBASA – CESPE 2009)

[96] O objetivo principal de um diagrama de pacotes é agrupar os pacotes em classes. Esse tipo de diagrama pode usar dependências.

(TJ/PE – FCC 2007)

[40] Diagrama de Pacotes descreve os pacotes ou pedaços do sistema, como o sistema é dividido em agrupamentos lógicos e mostra as dependências entre estes.

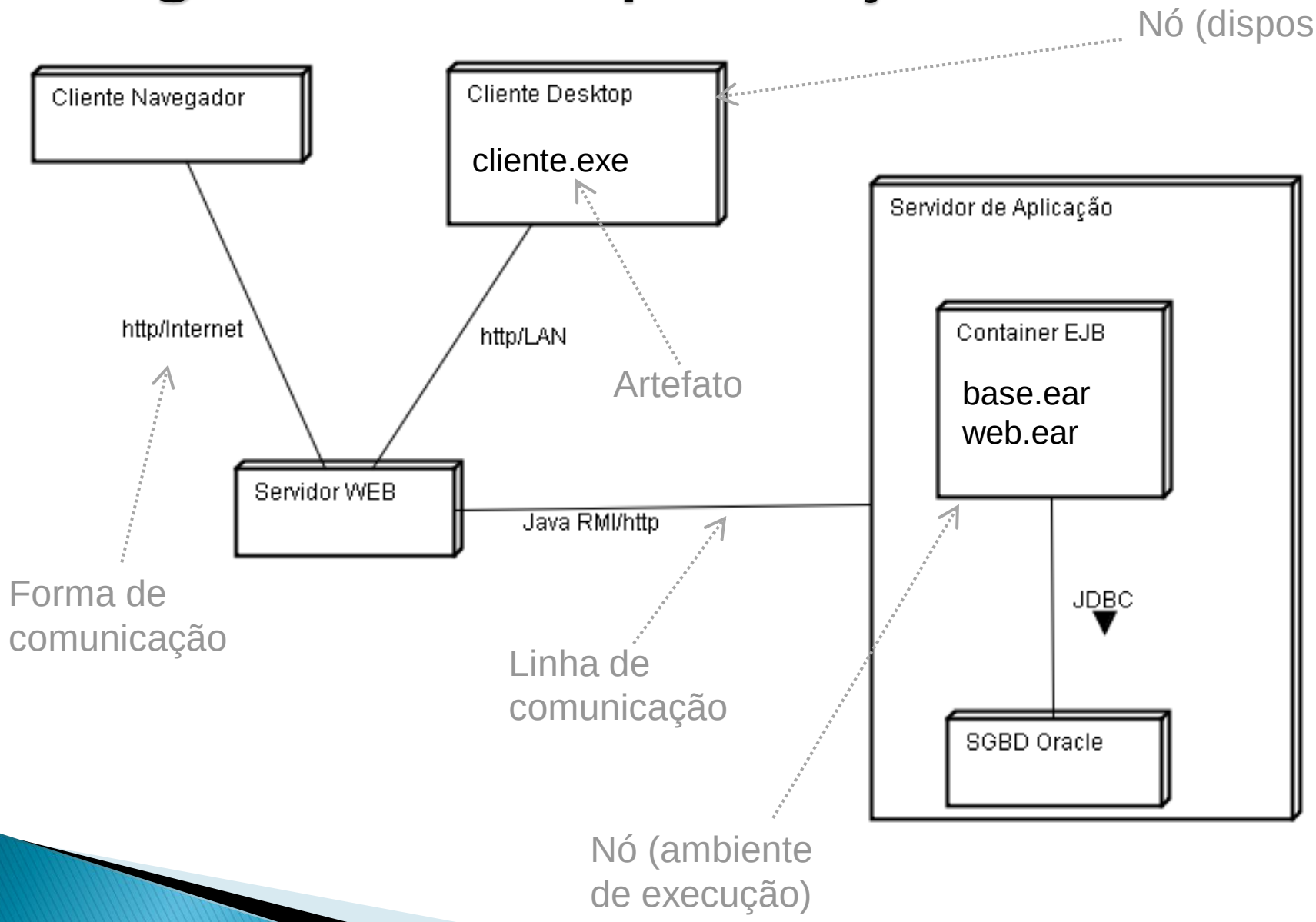
Diagramas Estruturais (estáticos)

- Diagrama de Classes
- Diagrama de Objetos
- Diagrama de Componentes
- Diagrama de Pacotes
- **Diagrama de Implantação**
- Diagrama de Estrutura Composta
- Diagrama de Perfis (UML 2.2)

Diagrama de Implantação

- Modela a configuração física do sistema, revelando que pedaços de software rodam em que equipamentos de hardware
- Inclui
  - Nós
    - Dispositivos (Hardware)
    - Ambientes de Execução
  - Artefatos
    - Código fonte, Código binário
    - Executáveis, etc.

Diagrama de Implantação



Exercícios [4]

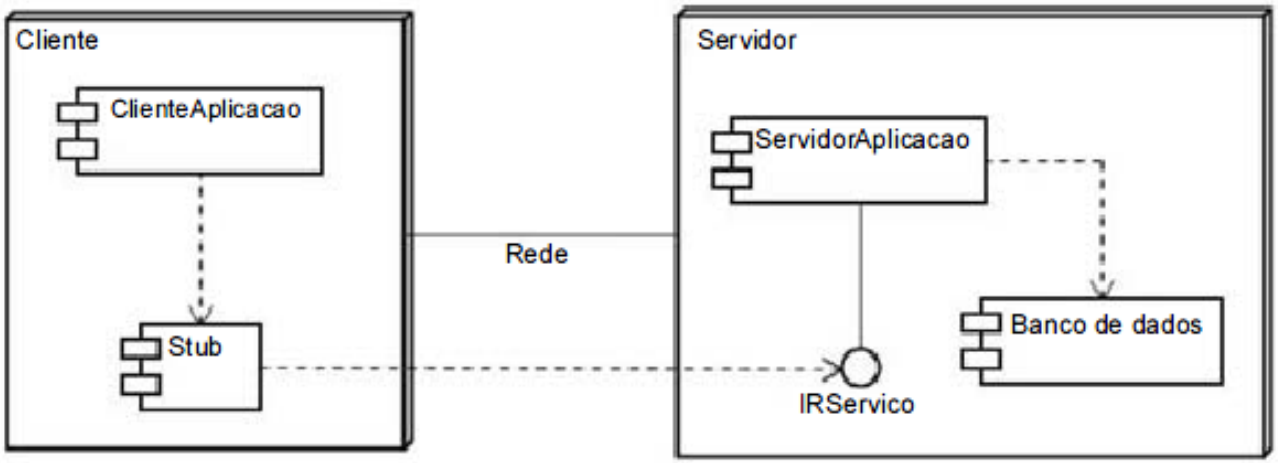


Figura III

(STJ – CESPE 2008)

[54] No diagrama da figura III, há dois nós interligados, que representam duas unidades computacionais; há cinco componentes distribuídos entre os nós; um destes implementa uma interface e um outro depende dessa interface; ClienteAplicacao depende de Stub; ServidorAplicacao depende de Banco de dados.

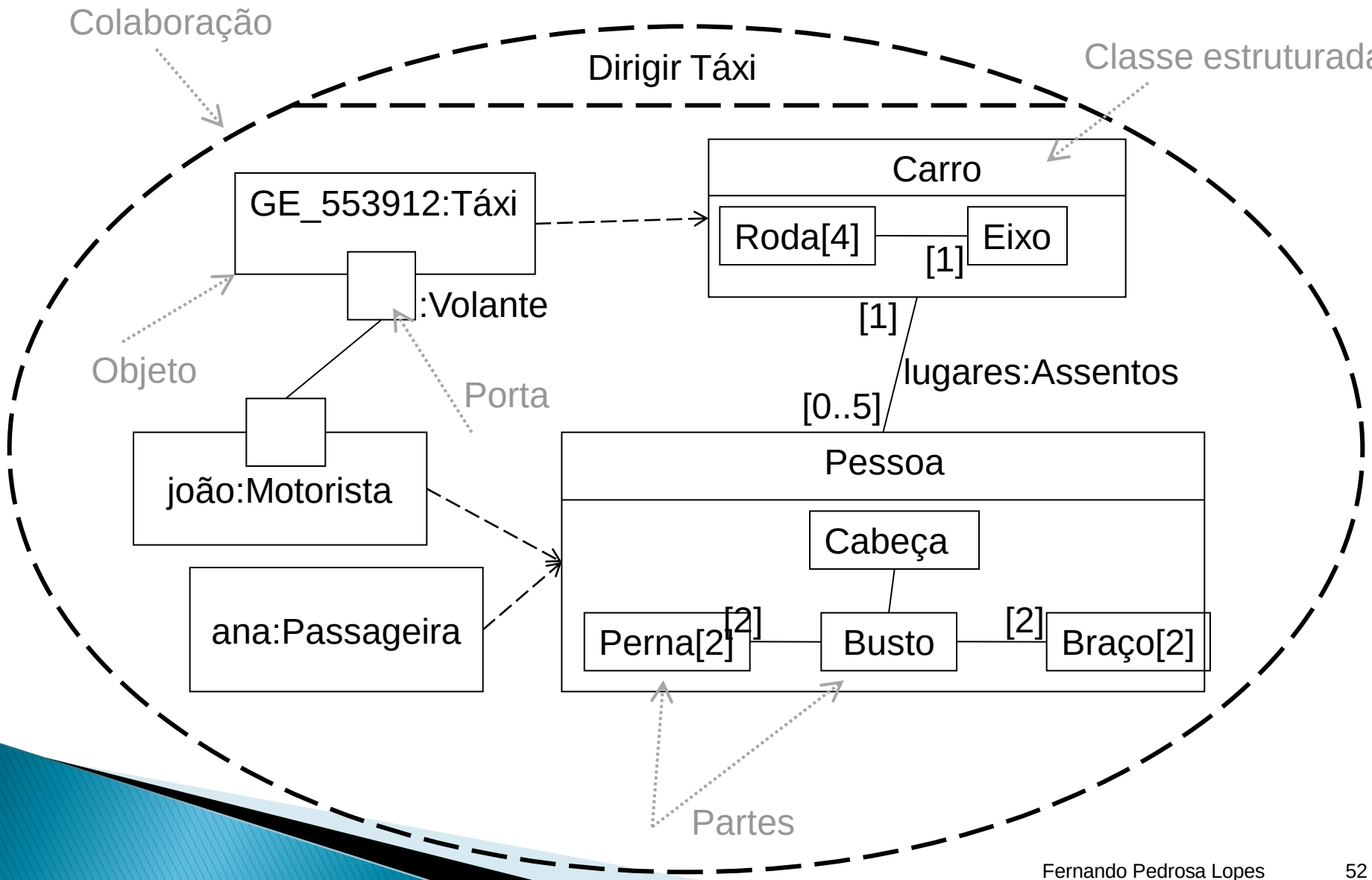
Diagramas Estruturais (estáticos)

- Diagrama de Classes
- Diagrama de Objetos
- Diagrama de Componentes
- Diagrama de Pacotes
- Diagrama de Implantação
- **Diagrama de Estrutura Composta**
- Diagrama de Perfis (UML 2.2)

Diagrama de Estrutura Composta

- É utilizado para modelar colaborações entre interfaces, objetos ou classes
- Pode ser usado para descrever
  - Estruturas de partes interconectadas
  - Estruturas de instâncias interconectadas
- **Parte**: representa o conjunto de uma ou mais instâncias contidas em outro elemento
- **Porta**: ponto de interação entre os elementos

Diagrama de Estrutura Composta



Exercícios [5]

(UNIPAMPA - CESPE 2010)

[105] Na UML 2.0, o diagrama de estrutura composta (composite structure diagram) descreve a estrutura interna de um classificador modelando as colaborações, no qual uma colaboração descreve uma visão de um conjunto de instâncias que cooperam entre si para executar uma função específica entre instâncias de classes, objetos ou interfaces.

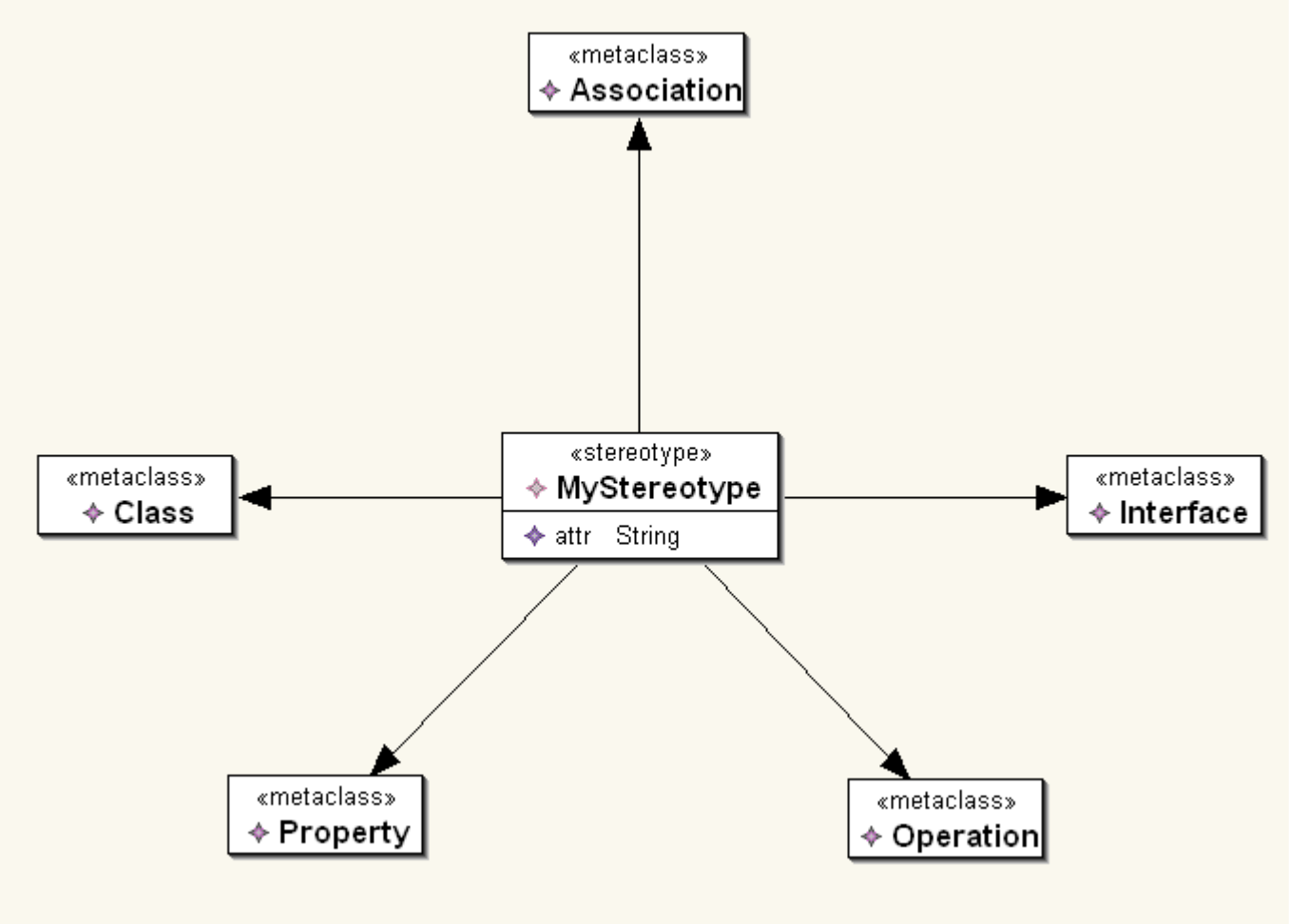
Diagramas Estruturais (estáticos)

- Diagrama de Classes
- Diagrama de Objetos
- Diagrama de Componentes
- Diagrama de Pacotes
- Diagrama de Implantação
- Diagrama de Estrutura Composta
- **Diagrama de Perfis (UML 2.2)**

Diagrama de Perfis (Profile Diagram)

- É um diagrama auxiliar que permite definir tipos padronizados de estereótipos, valores rotulados e restrições
- A UML define o mecanismo de perfis como um “mecanismo leve de extensão” da linguagem
- Permite adaptar os modelos UML para diferentes plataformas e domínios

Diagrama de Perfis (Profile Diagram)



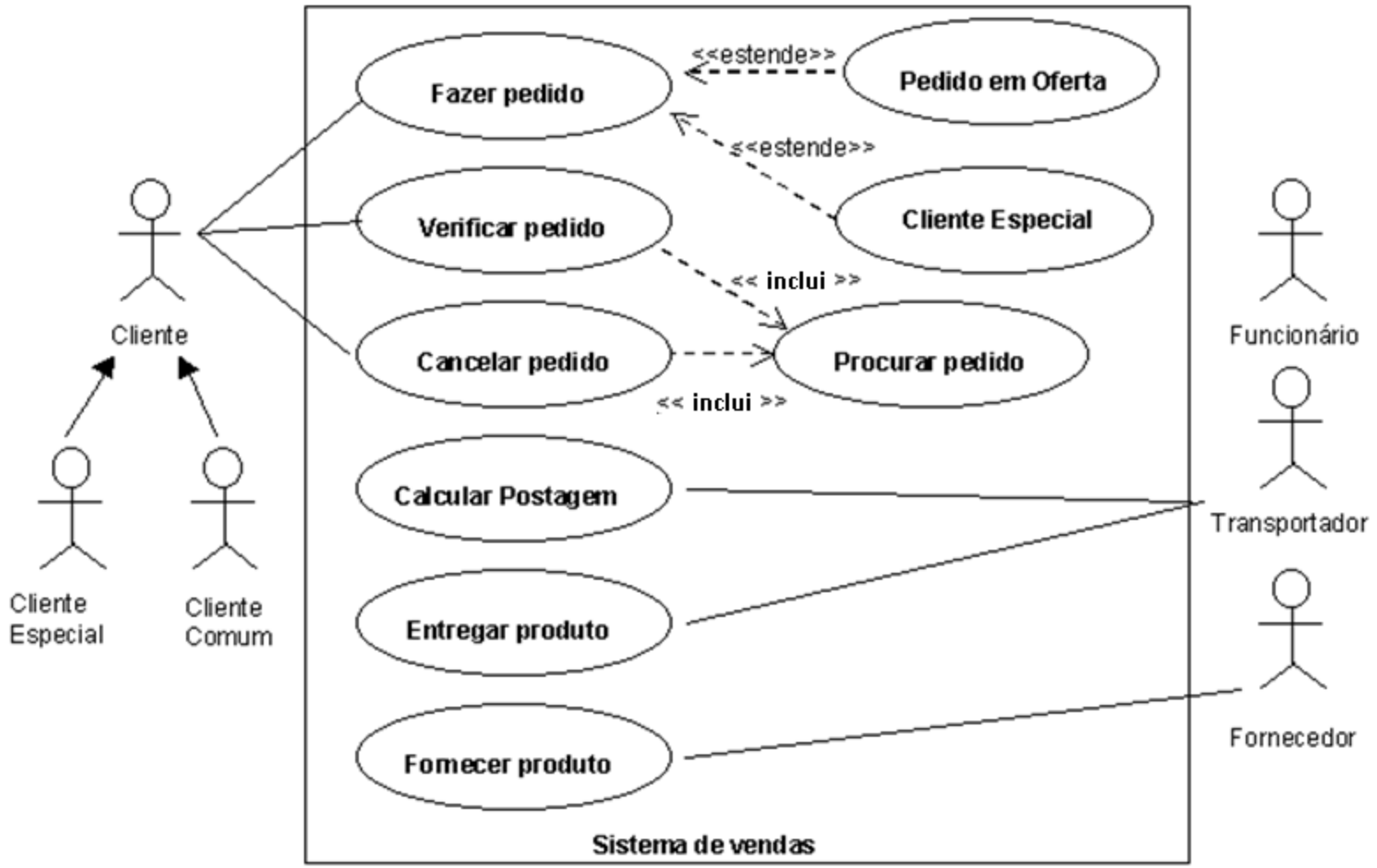
Diagramas Comportamentais (dinâmicos)

- **Diagrama de Casos de Uso**
- Diagrama de Atividade
- Diagrama de Máquina de Estados
- Diagramas de Interação
  - Diagrama de Sequência
  - Diagrama de Comunicação
  - Diagrama de Tempo
  - Diagrama de Interação Geral

Diagrama de Casos de Uso

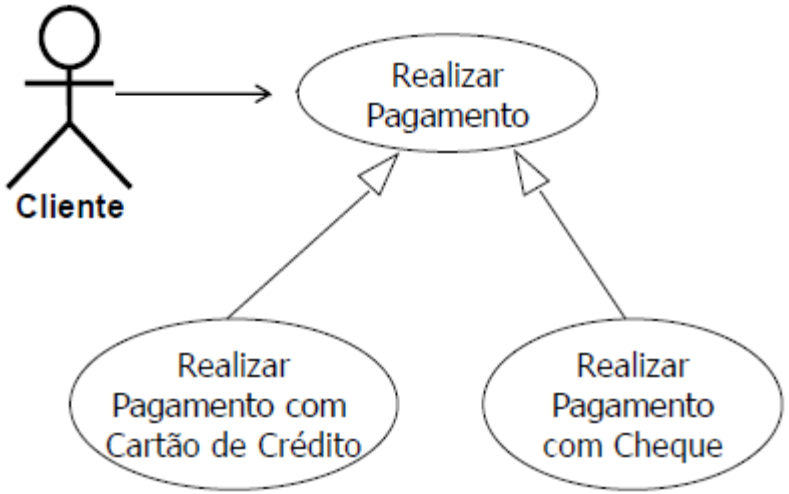
- Contém um conjunto de casos de uso e modela interações entre
  - Atores e o sistema
  - O próprio sistema
- Descreve um conjunto de cenários
- Captura os requisitos do usuário
- Delimita o escopo do sistema

Diagrama de Caso de Uso



Generalizações entre Casos de Uso

- O filho herda o comportamento do pai, podendo adicionar e redefinir passos em pontos arbitrários do comportamento original



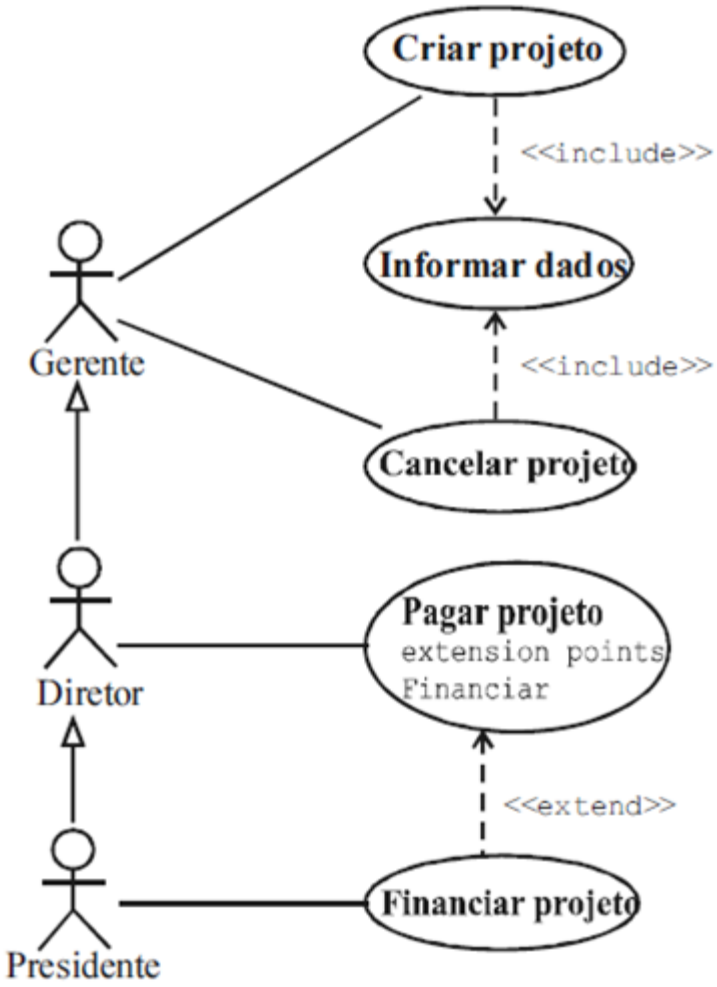
Inclusão e Extensão

- Inclusão
  - Use quando o mesmo comportamento se repete em mais de um Caso de Uso e o processo de realizar X **sempre** envolve realizar Y pelo menos uma vez
- Extensão
  - Use quando você quiser modelar um comportamento **opcional** de um Caso de Uso

Generalização entre Atores

- Use quando um ator (filho) é um **tipo de** outro ator mais genérico (pai)

- Exemplo:



Tipos de Casos de Uso

- Concreto
  - É iniciado por um ator e constitui um fluxo completo de eventos
- Abstrato: nunca é instanciado diretamente
  - Casos de Uso abstratos geralmente são:
    - Incluídos em outros Casos de Uso
    - Estendidos de outros Casos de Uso
    - Generalizações de outros Casos de Uso
- Atores “enxergam” apenas casos de uso concretos

Exercícios [6]

(EMBASA – CESPE 2009)

[95] Um diagrama de casos de uso descreve um cenário que mostra as funcionalidades do sistema do ponto de vista do usuário. É comum o uso de atores nesse diagrama.

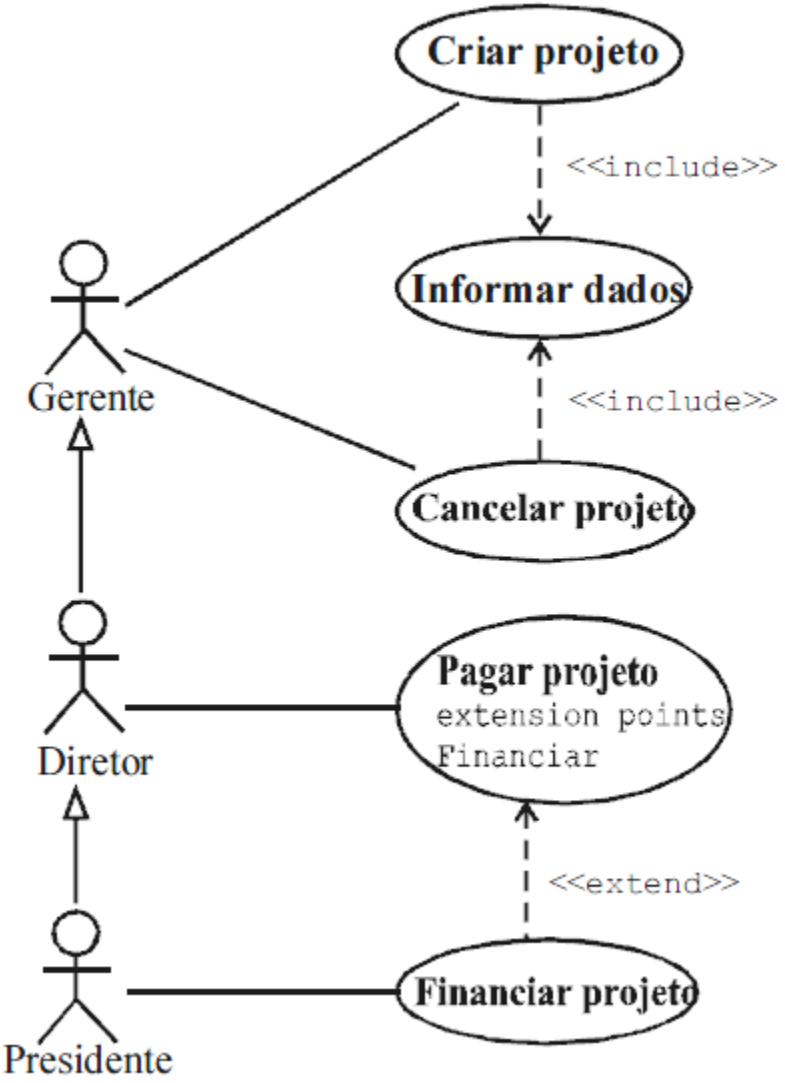
(MPE/AM – CESPE 2008)

[85] Em um diagrama de casos de uso da UML, um ator é definido como um usuário humano do sistema.

Exercícios [6]

(MPE/RR – CESPE 2008)

[87] No diagrama UML ao lado, o ator Presidente está relacionado ao caso de uso Criar projeto; o caso de uso informar dados contém comportamento comum a dois casos de uso; o caso de uso Pagar projeto estende o comportamento Financiar projeto e Cancelar projeto é abstrato.



Diagramas Comportamentais (dinâmicos)

- Diagrama de Casos de Uso
- **Diagrama de Atividade**
- Diagrama de Máquina de Estados
- Diagramas de Interação
  - Diagrama de Sequência
  - Diagrama de Comunicação
  - Diagrama de Tempo
  - Diagrama de Interação Geral

Diagrama de Atividade

- Descreve lógicas de procedimento, processos de negócio e fluxos de trabalho
- Permite que seja mostrado que entidade é responsável por cada ação no diagrama, com uso de raias (*swimlanes*)
  - Quem faz o quê?

Diagrama de Atividade

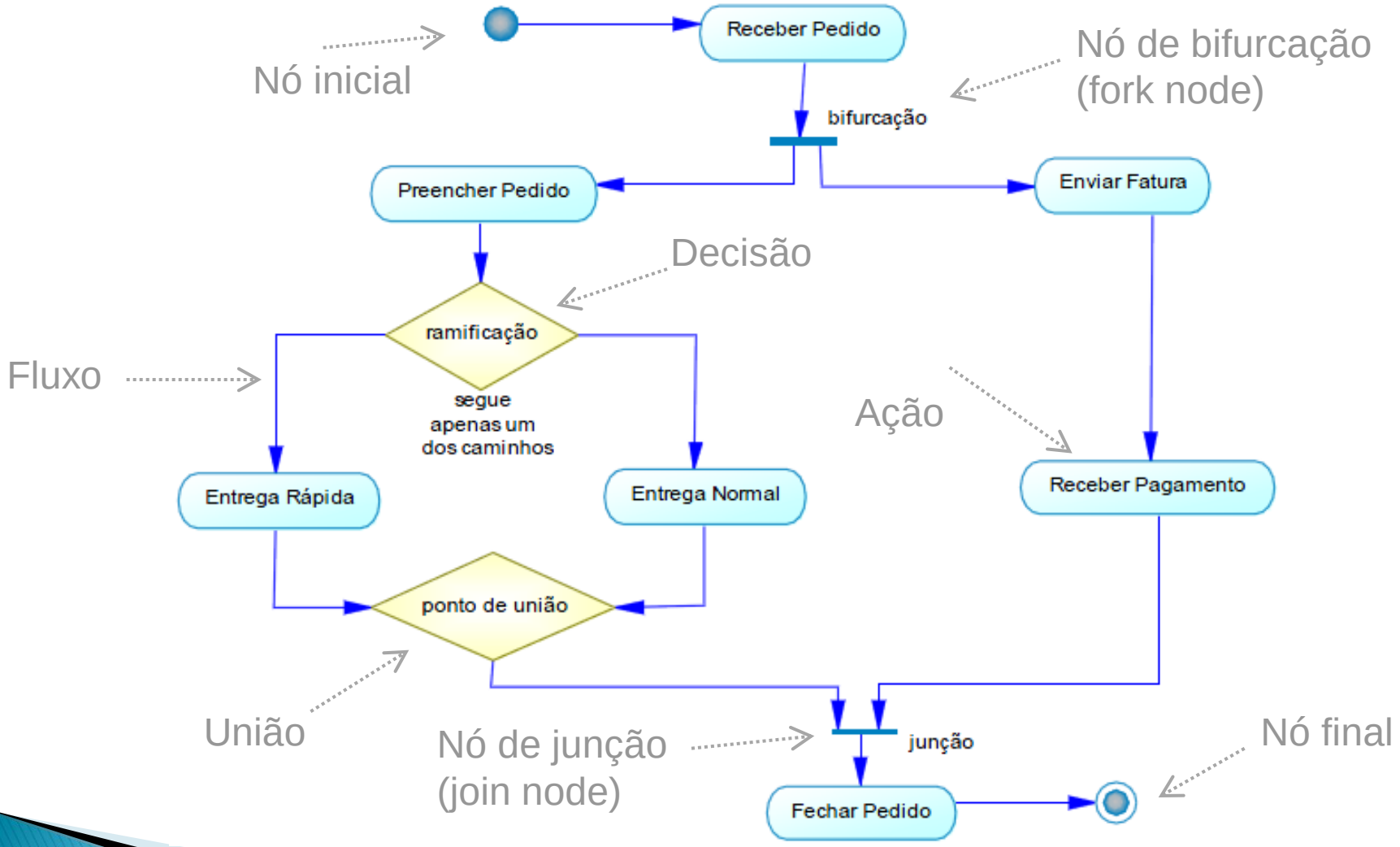
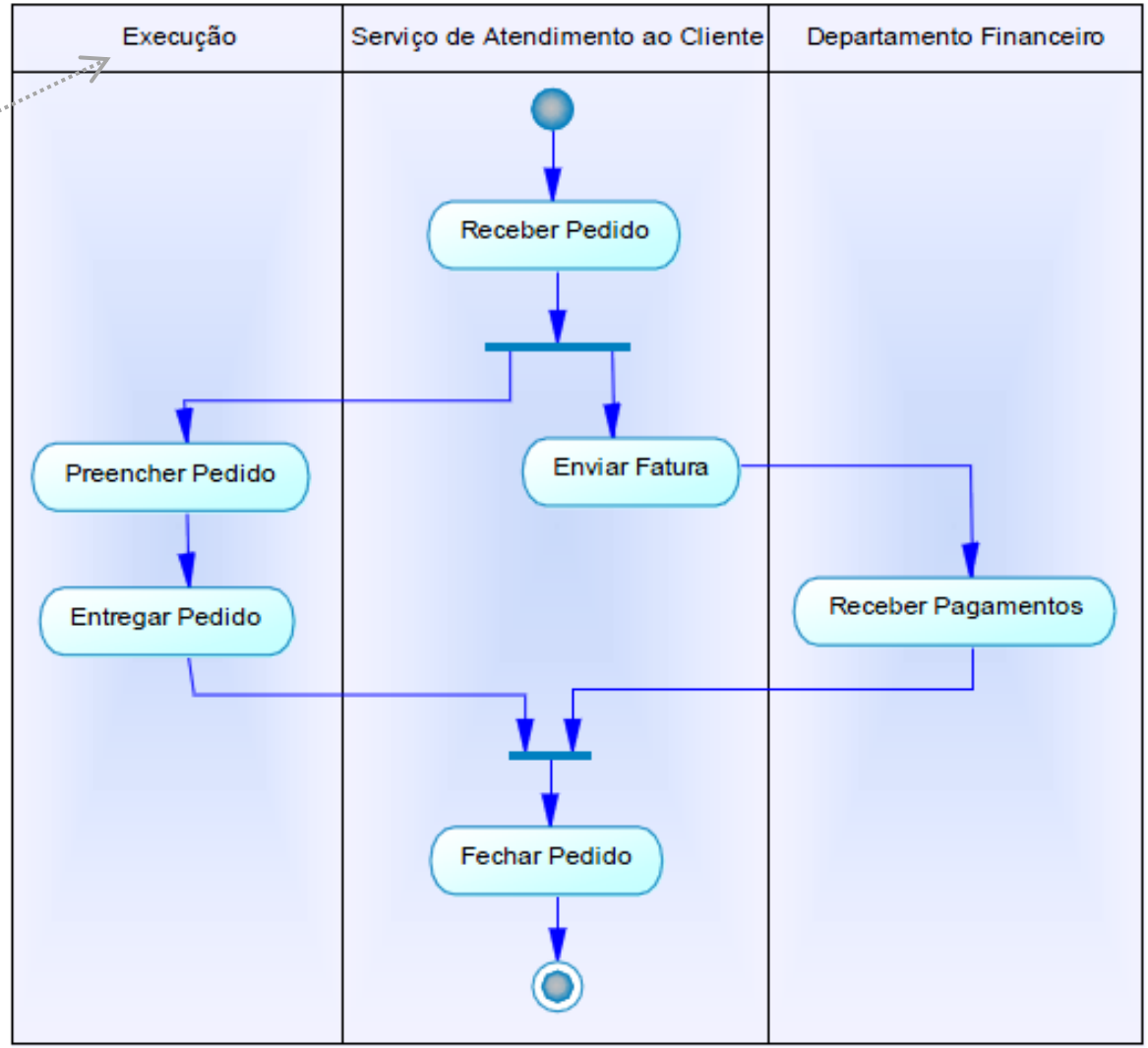


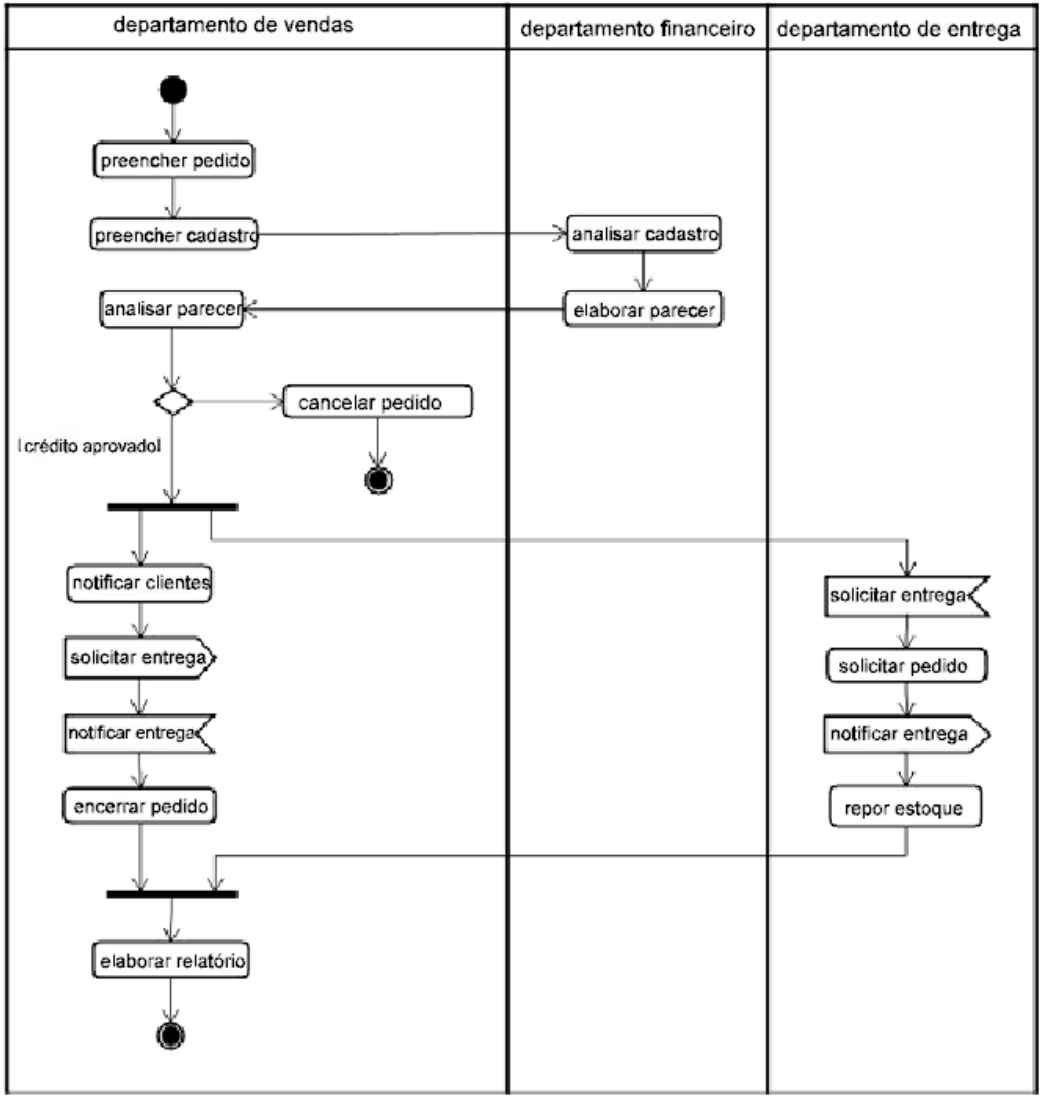
Diagrama de Atividade (swimlanes)

Unidades organizacionais



Exercícios [7]

(PETROBRAS – CESPE 2007)
Com referência ao diagrama de atividades UML ao lado, julgue os itens a seguir.



Exercícios [7]

[88] No diagrama, há duas raias, um estado inicial e dois finais. Por estarem em raias distintas, a atividade Preencher cadastro pode ser realizada em paralelo à atividade Analisar cadastro. Na decisão representada pelo losango, apenas uma condição de guarda é especificada, o que torna o diagrama incorreto.

[89] A atividade Notificar cliente pode ser executada em paralelo à atividade Entregar produto, mas a atividade Encerrar pedido não pode ser executada em paralelo à atividade Repor estoque. A atividade Elaborar relatório será executada após ser concluída a atividade Encerrar pedido ou Repor estoque.

Diagramas Comportamentais (dinâmicos)

- Diagrama de Casos de Uso
- Diagrama de Atividade
- **Diagrama de Máquina de Estados**
- Diagramas de Interação
  - Diagrama de Sequência
  - Diagrama de Comunicação
  - Diagrama de Tempo
  - Diagrama de Interação Geral

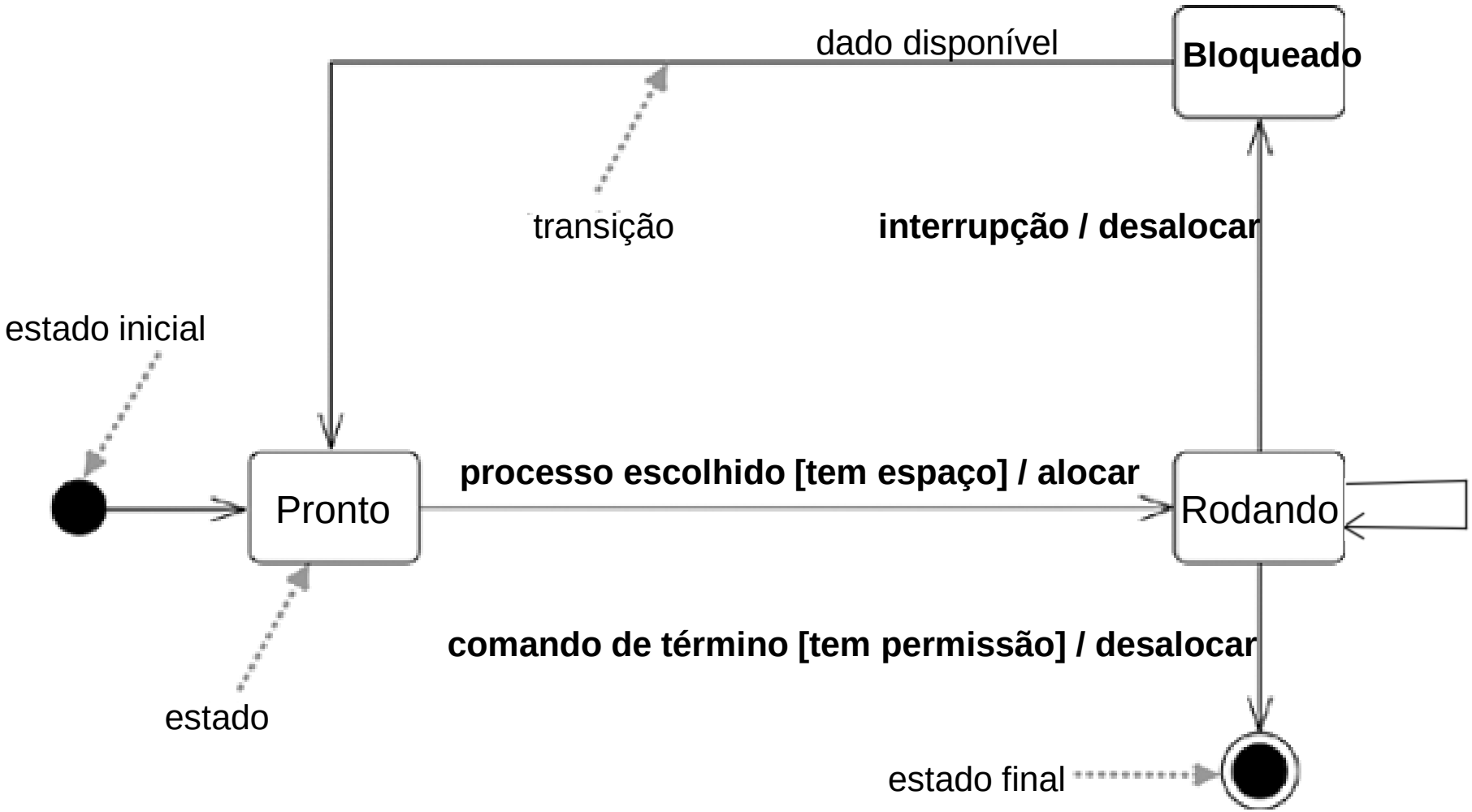
Diagrama de Máquina de Estados

- Mostra os vários estados possíveis por quais um objeto pode passar
- Um objeto muda de estado quando acontece algum evento interno ou externo ao sistema
- Através da análise das transições entre os estados, pode-se prever todas as possíveis operações realizadas, em função de eventos que podem ocorrer

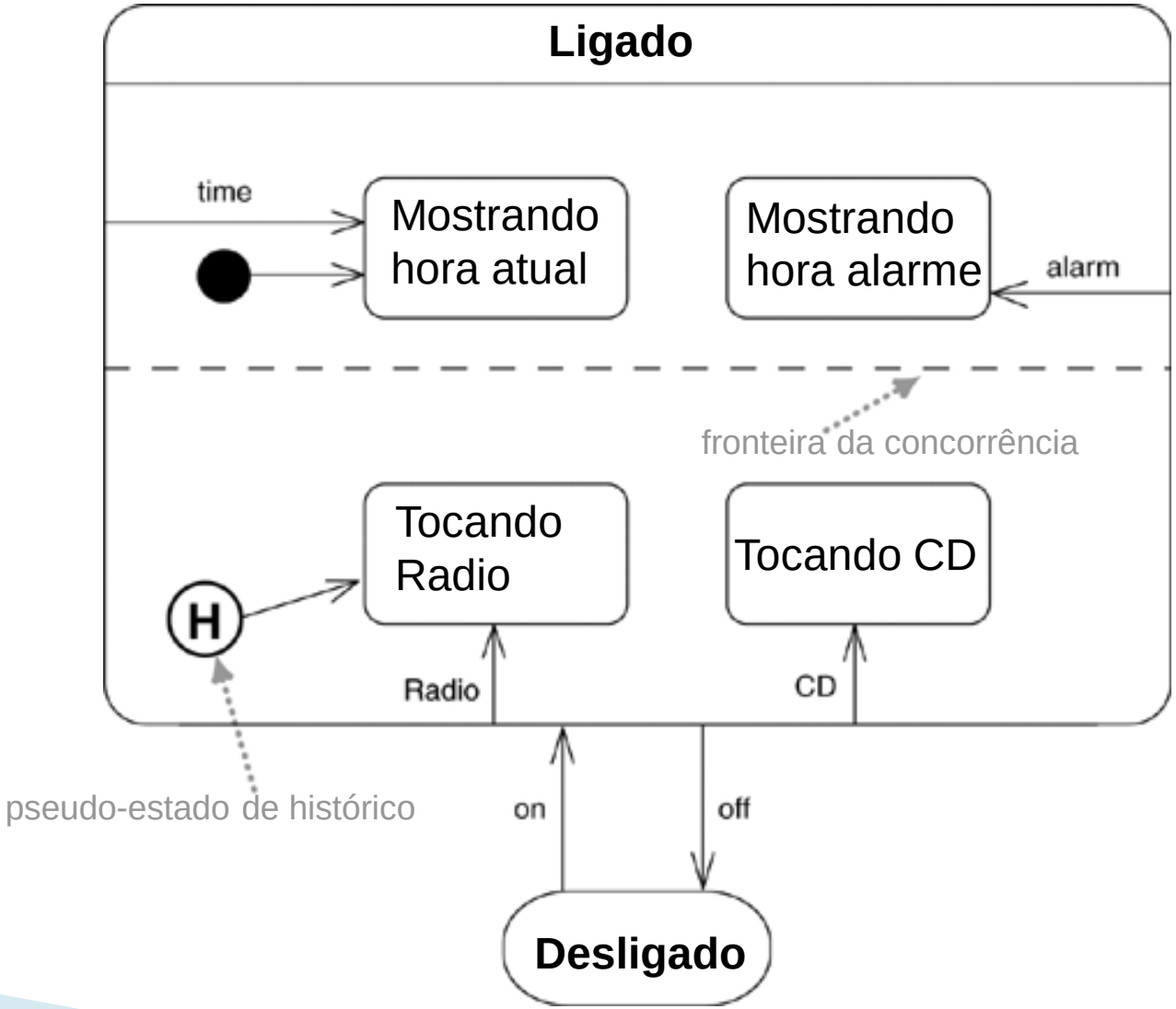
Elementos

- Estados
  - Situações na vida de um objeto na qual ele satisfaz uma condição ou realiza alguma atividade
- Transições
  - Estados são associados através de transições
  - Transições têm eventos associados
    - Sintaxe: **evento [condição]/ação**
- Ações
  - Ao passar de um estado para o outro o objeto pode realizar ações
- Atividades
  - Executadas durante um estado

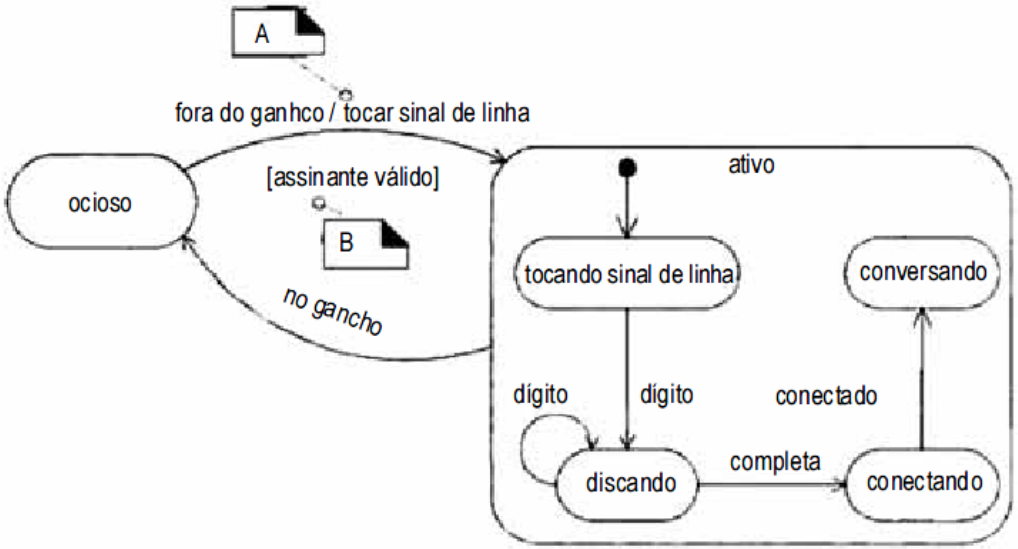
Ex: Escalonamento de Processos



Estados aninhados e concorrentes



Exercícios [8]



(EMBASA – CESPE 2009)

A figura acima é um exemplo de diagrama de transição de estados, que permite modelar como o sistema responde a eventos internos e externos, especificando o que acontece quando o evento ocorre. Ele é útil para modelar o comportamento de sistemas de tempo real, já que tais sistemas lidam com estímulos do ambiente. A respeito desse assunto e da figura acima, julgue os próximos itens.

Exercícios [8]

[73] É possível criar um diagrama de transição de estados que descreva o ciclo de vida de um objeto em níveis de detalhe arbitrariamente simples ou complexos, dependendo das necessidades, pois não há a obrigação de ilustrar todos os eventos possíveis.

[74] Na figura, A associa-se a uma ação de guarda, e B, a uma ação de transição.

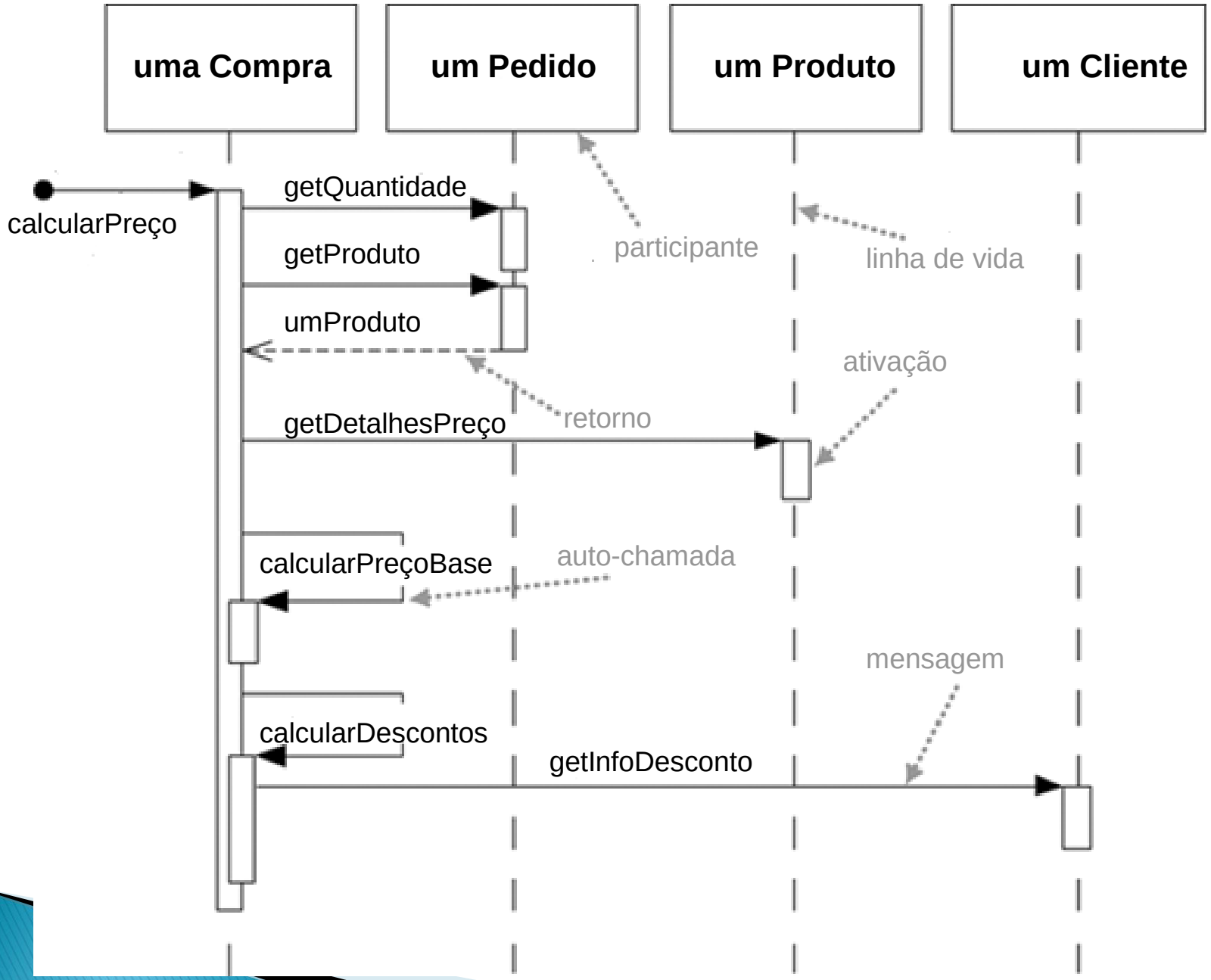
[97] Um diagrama de estado é capaz de mostrar os estados possíveis de um objeto. Além disso, pode mostrar as transações responsáveis pelas suas mudanças de estado.

Diagramas Comportamentais (dinâmicos)

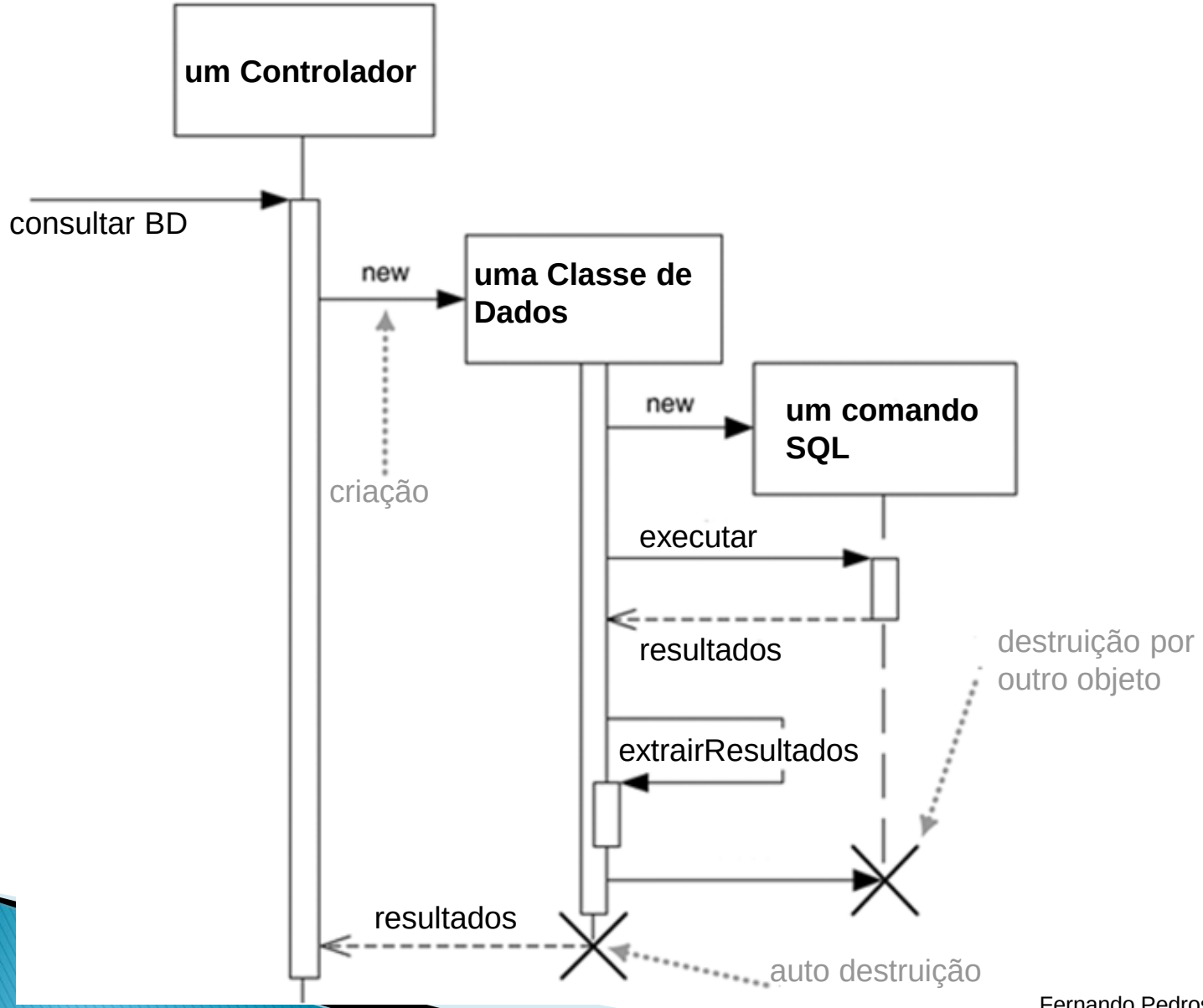
- Diagrama de Casos de Uso
- Diagrama de Atividade
- Diagrama de Máquina de Estados
- **Diagramas de Interação**
  - **Diagrama de Sequência**
  - Diagrama de Comunicação
  - Diagrama de Tempo
  - Diagrama de Interação Geral

Diagrama de Sequência

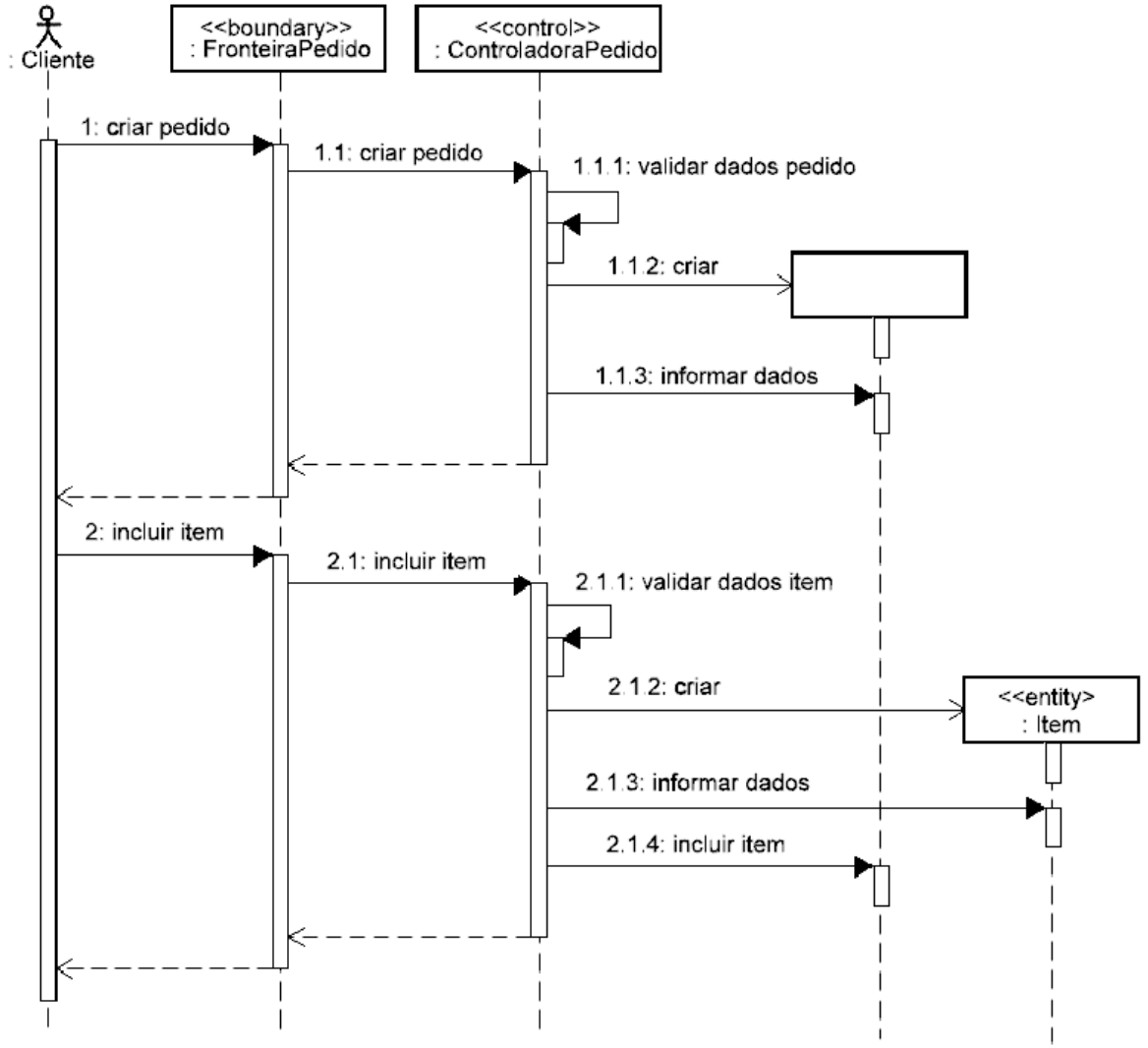
- Captura o comportamento de um determinado cenário
- Mostra os objetos e as mensagens trocadas entre eles
- Enfatiza a **ordem temporal** das mensagens
- É o diagrama mais utilizado na etapa de Projeto OO (solucionar o problema)


uma Compra
um Pedido
um Produto
um Cliente
calcularPreço
getQuantidade
getProduto
umProduto
participante
linha de vida
ativação
retorno
getDetalhesPreço
calcularPreçoBase
auto-chamada
mensagem
calcularDescontos
getInfoDesconto

Criação e destruição de objetos



Exercícios [9]



Exercícios [9]

(PETROBRAS - CESPE 2007)
Julgue os itens a seguir, relativos ao diagrama de seqüência UML apresentado acima.

[91] Dois objetos existiam antes da interação e dois foram criados durante a interação. As setas da instância de ControladoraPedido para a instância de FronteiraPedido são retornos de mensagens. Um dos objetos tem nome Pedido e outro, Item. No diagrama, encontram-se representadas as linhas da vida dos objetos e as áreas de ativação das mensagens.

[92] São assíncronas as mensagens da instância de FronteiraPedido para a de ControladoraPedido. Há um erro no diagrama, pois uma instância de uma classe não pode enviar mensagens para ela mesma.

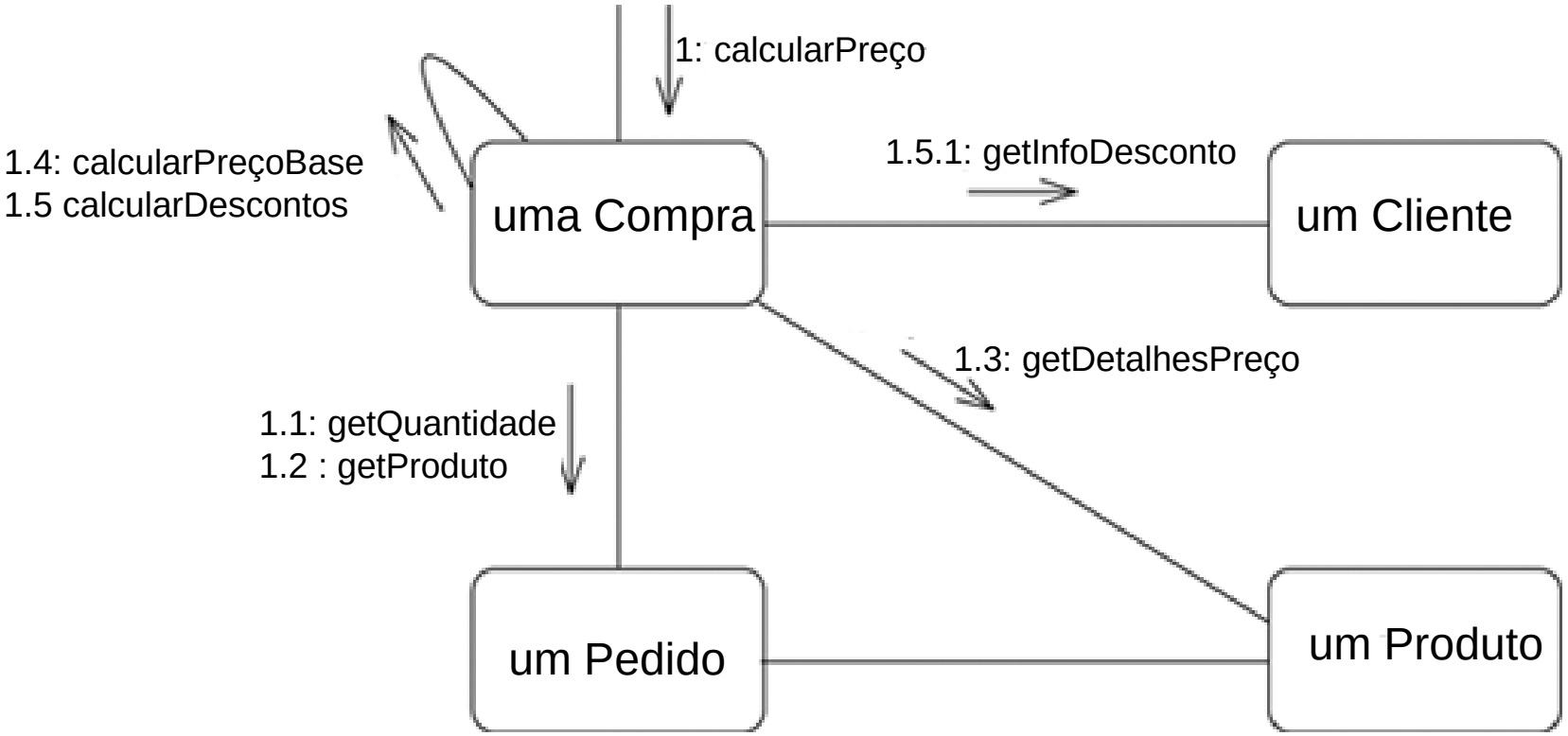
Diagramas Comportamentais (dinâmicos)

- Diagrama de Casos de Uso
- Diagrama de Atividade
- Diagrama de Máquina de Estados
- **Diagramas de Interação**
  - Diagrama de Sequência
  - **Diagrama de Comunicação**
  - Diagrama de Tempo
  - Diagrama de Interação Geral

Diagrama de Comunicação

- Captura o comportamento de um determinado cenário
- Mostra os objetos e as mensagens trocadas entre eles
- Enfatiza a **ordem estrutural** das mensagens (relacionamentos entre objetos)
- É equivalente ao diagrama de sequência

Diagrama de Comunicação



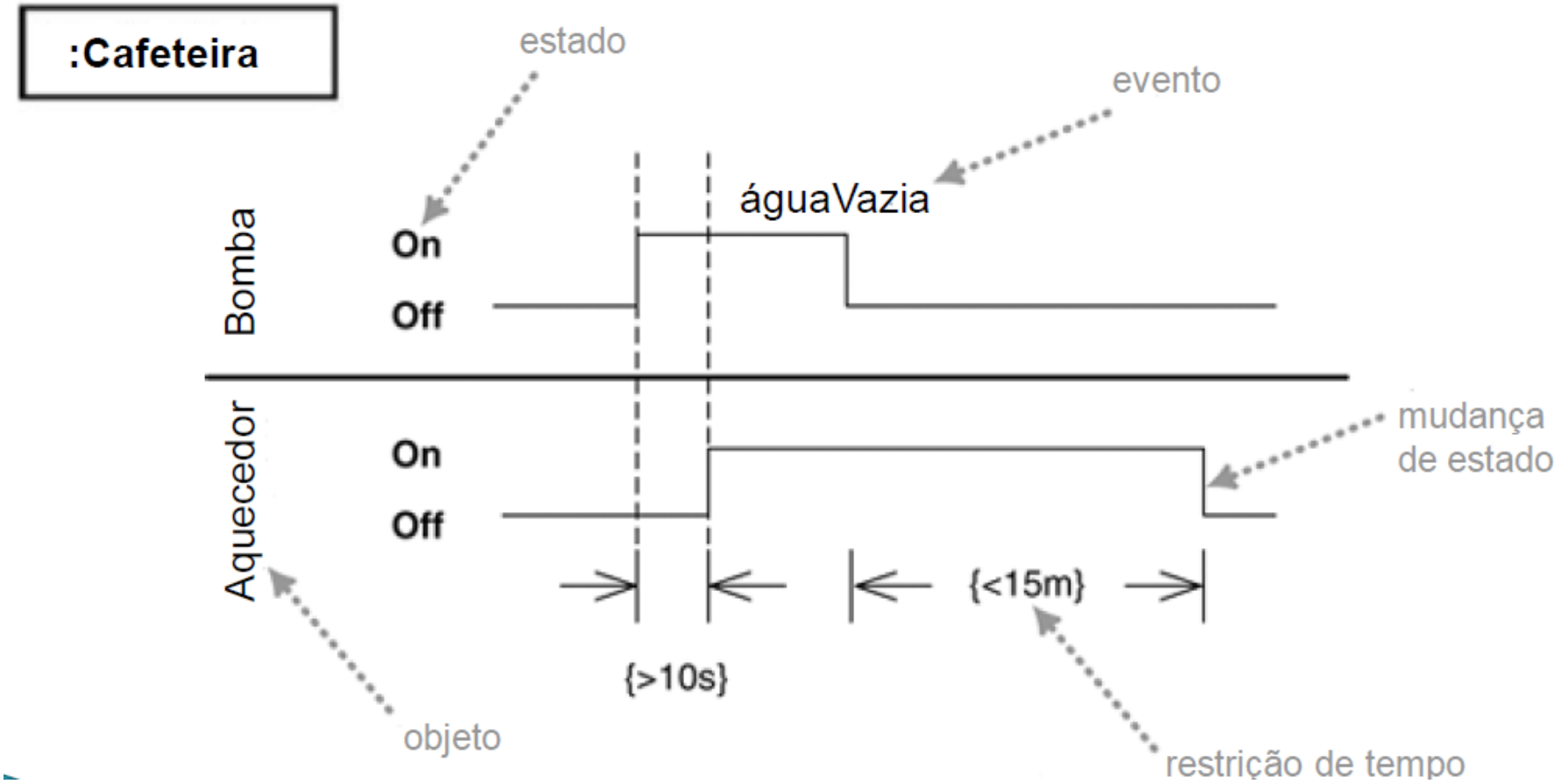
Diagramas Comportamentais (dinâmicos)

- Diagrama de Casos de Uso
- Diagrama de Atividade
- Diagrama de Máquina de Estados
- **Diagramas de Interação**
  - Diagrama de Sequência
  - Diagrama de Comunicação
  - **Diagrama de Tempo**
  - Diagrama de Interação Geral

Diagrama de Tempo

- Captura o comportamento de objetos ao longo do tempo e a duração na qual eles permanecem em determinados estados
- O foco se dá nas **restrições de tempo** das interações
- É uma mistura entre o diagrama de sequência e o diagrama de máquina de estados

Diagrama de Tempo



Diagramas Comportamentais (dinâmicos)

- Diagrama de Casos de Uso
- Diagrama de Atividade
- Diagrama de Máquina de Estados
- **Diagramas de Interação**
  - Diagrama de Sequência
  - Diagrama de Comunicação
  - Diagrama de Tempo
  - **Diagrama de Interação Geral**

Diagrama de Interação Geral

- Fornece uma visão geral do controle de fluxo entre objetos
- É uma mistura entre diagramas de sequência e diagramas de atividade

No exemplo, se o Cliente for externo, os dados são buscados de um XML. Se for interno, os dados são buscados de um banco de dados. A seqüência destes dois fluxos é detalhada. Ao final, é gerado um relatório

Diagrama de Interação Geral

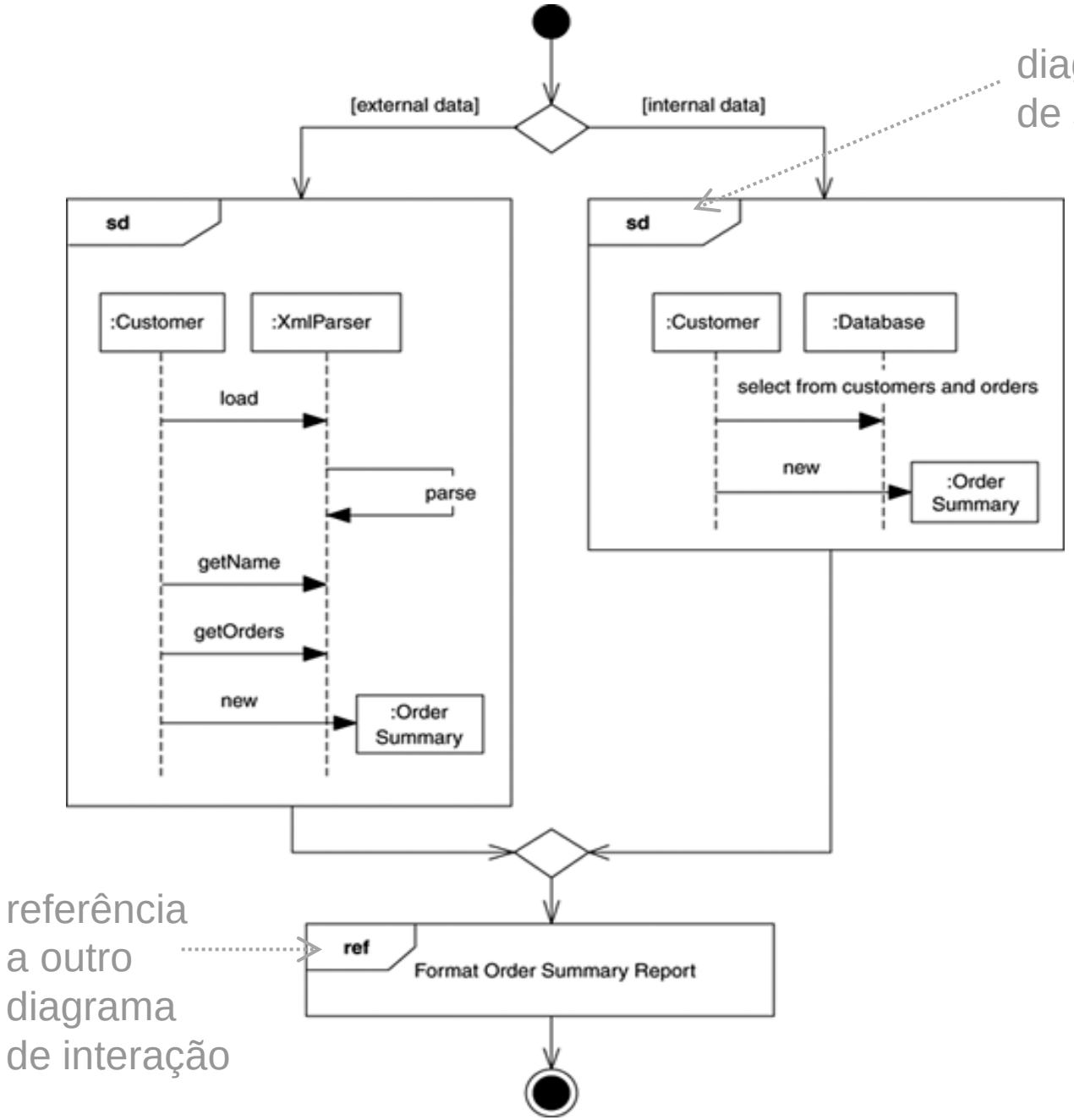


diagrama de sequência

referência a outro diagrama de interação

Object Constraint Language

- Linguagem que faz parte da UML e tem o objetivo de desenvolver modelos mais precisos
- Uma **restrição** (constraint) atua sobre um ou mais valores de um modelo orientado a objetos
- Vantagens
  - Modelos mais completos, consistentes e precisos
  - Comunicação sem ambigüidade
  - Sintaxe e semântica formais

Object Constraint Language

- Exemplos de restrições (regras de um sistema de Universidade)
  - “A avaliação de supervisores acadêmicos deve ser maior que a nota dos seus supervisionados”
  - “A bolsa escolar dos alunos depende da sua avaliação acadêmica”
- Estas regras podem ser escritas em OCL
- E podem ser transformadas em
  - Código
  - *Scripts* de bancos de dados
  - Outros modelos, etc.

Gabaritos dos Exercícios

- [1] 78 E, 94 E, 101 E, 31 D
- [2] 88 C, 89 C, 108 E, 109 C, 110 X (E)
- [3] 106 E, 40 C, 96 E, 40 C
- [4] 54 E
- [5] 105 C
- [6] 95 C, 85 E, 87 E
- [7] 88 E, 89 E
- [8] 73 C, 74 E, 97 C
- [9] 91 E, 92 E

FIM