Arquitetura de Software

Diagramas UML - Especificando a Arquitetura de Software

Nairon Neri Silva

Sumário

- Especificação da Arquitetura
- Modelagem Estrutural
- Diagrama de Pacotes
- Diagrama de Componentes
  - Portas
  - Componentes e Conectores

Especificação da Arquitetura

- Uma importante forma de “comunicar” a arquitetura é através de modelos e diagramas

- Todos os diagramas da UML podem ser utilizados para especificar aspectos da arquitetura de um sistema

Especificação da Arquitetura

- Modelagem estrutural “avançada”
  - Diagrama de Pacotes
  - Diagrama de Componentes

- Modelagem de comportamento
  - Diagrama de Casos de Uso

- Modelagem da Arquitetura
  - Diagrama de Implantação
  - Padrões e Frameworks

Modelagem Estrutural

Diagrama de Pacotes

Diagrama de Pacotes

- Diagramas de pacotes são diagramas estruturais usados para mostrar, em uma forma de pacotes, a organização e disposição de vários elementos de modelos

- Auxilia a gerência de grande quantidade de classes, interfaces, componentes, nós, diagramas e outros elementos

- Na UML pacote é um mecanismo de propósito geral para a organização de elementos de modelagem em grupos

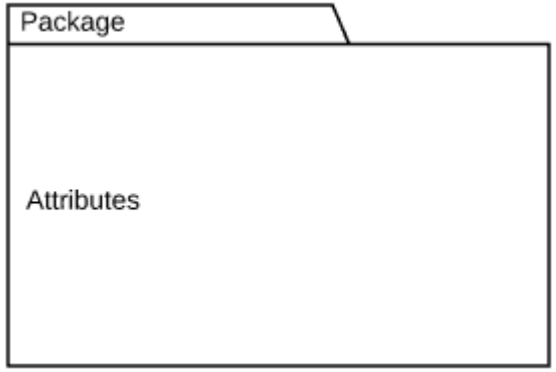
Pacotes

- Os pacotes...
  - Agrupam elementos próximos semanticamente
  - Podem ser hierarquicamente organizados
  - Podem ser utilizados para armazenar código e outros elementos (p.ex.: documentação do sistema, testes, etc.)

Benefícios dos diagramas de pacotes

- O diagrama fornece uma visão clara da estrutura hierárquica dos variados elementos UML dentro de um determinado sistema
- Esses diagramas podem simplificar diagramas de classes complexos, criando elementos visuais organizados.
- Eles oferecem uma ótima visibilidade geral de projetos e sistemas de grande escala.
- Os elementos visuais podem ser atualizados com facilidade conforme a evolução dos sistemas e projetos.

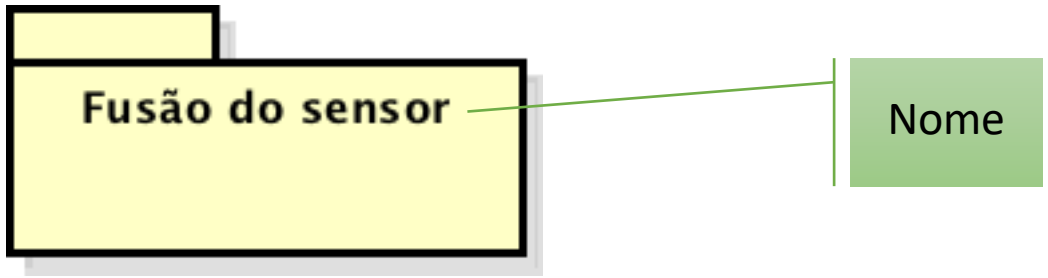
Componentes básicos



Pacote: Agrupa elementos com base em dados, comportamentos ou interação do usuários

Dependência: Mostra a relação entre um elemento e outro

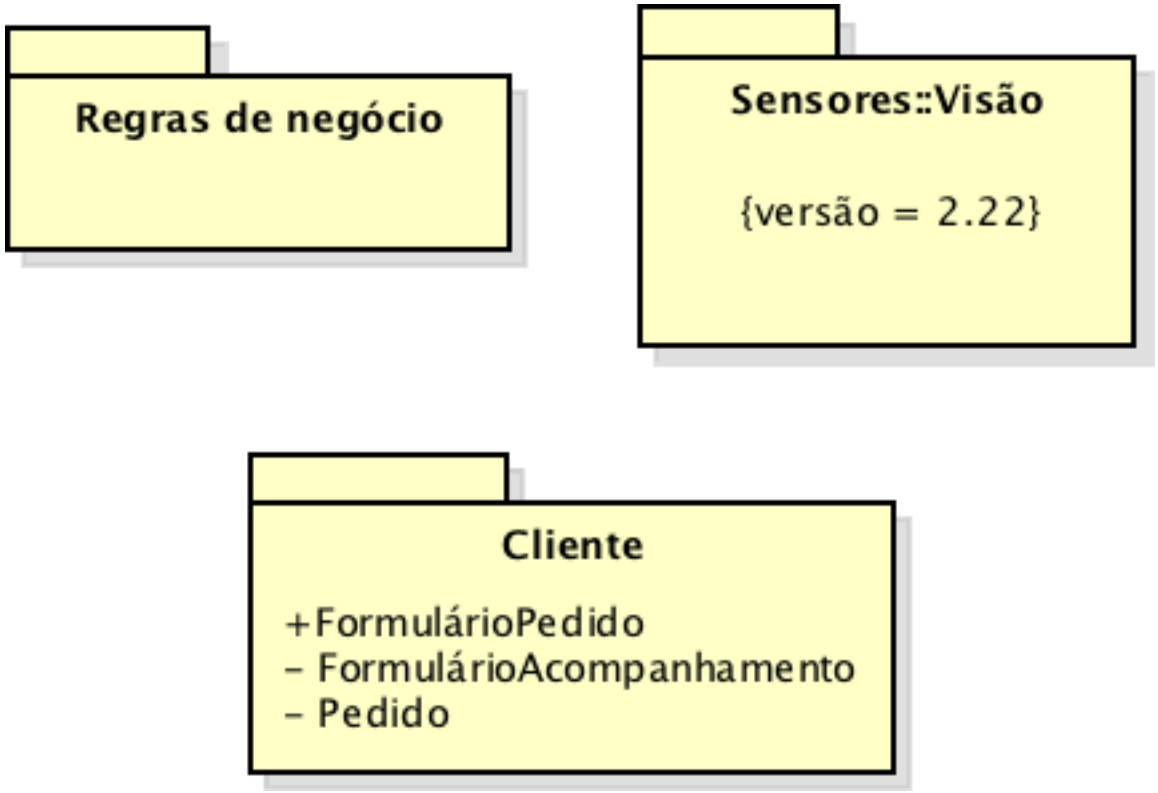
Representação de Pacotes



- Nome do pacote:
  - Nome simples
  - Nome qualificado → apresenta como prefixo o nome do pacote que o contém, separados por dois-pontos duplos "::"

Representação de Pacotes

- Exemplos:



Representação de Pacotes

- Um pacote pode conter:
  - Classes
  - Interfaces
  - Componentes
  - Nós
  - Colaborações
  - Casos de uso
  - Diagramas
  - Além de “outros pacotes”

Notações de dependência

- Os diagramas de pacotes são usados, em parte, para mostrar as dependências de importação e acesso entre pacotes, classes, componentes e outros elementos nomeados dentro do seu sistema.

- Cada dependência é renderizada como uma linha de conexão com uma seta, representando o tipo de relacionamento entre os dois ou mais elementos.

- Os tipos principais de dependências são: **Acesso** e **Importação**

Notações de dependência

Dependência de **acesso** indica que um pacote requer assistência das funções de outro pacote.

Exemplo:

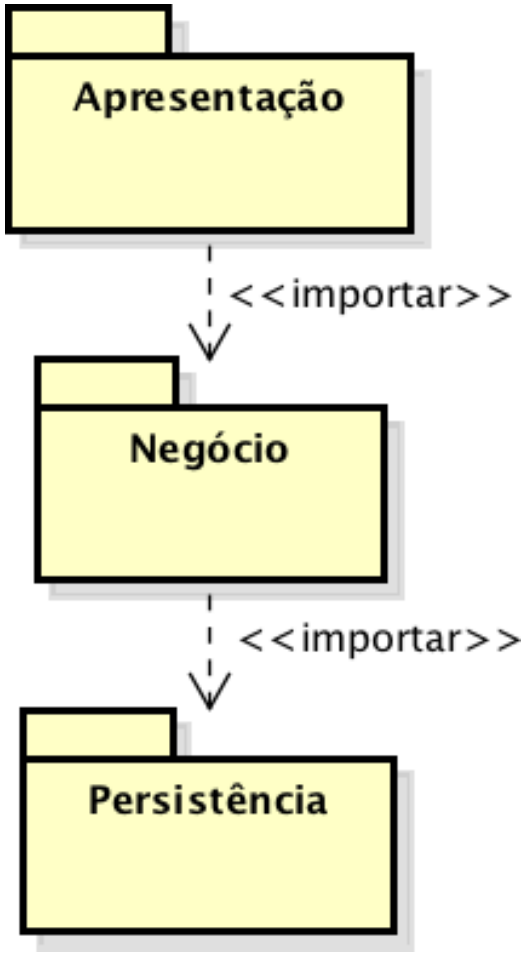
Notações de dependência

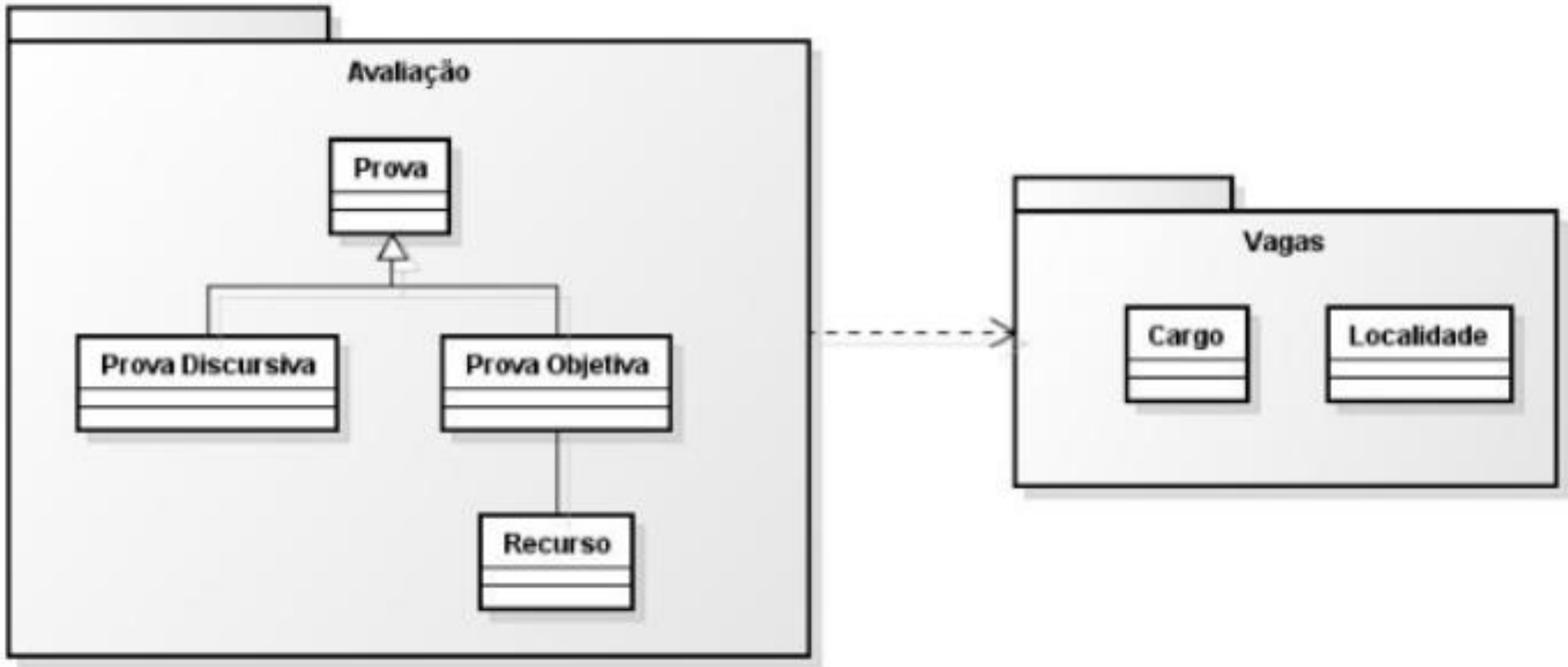
Dependência de **importação** indica que a funcionalidade foi importada de um pacote para outro.

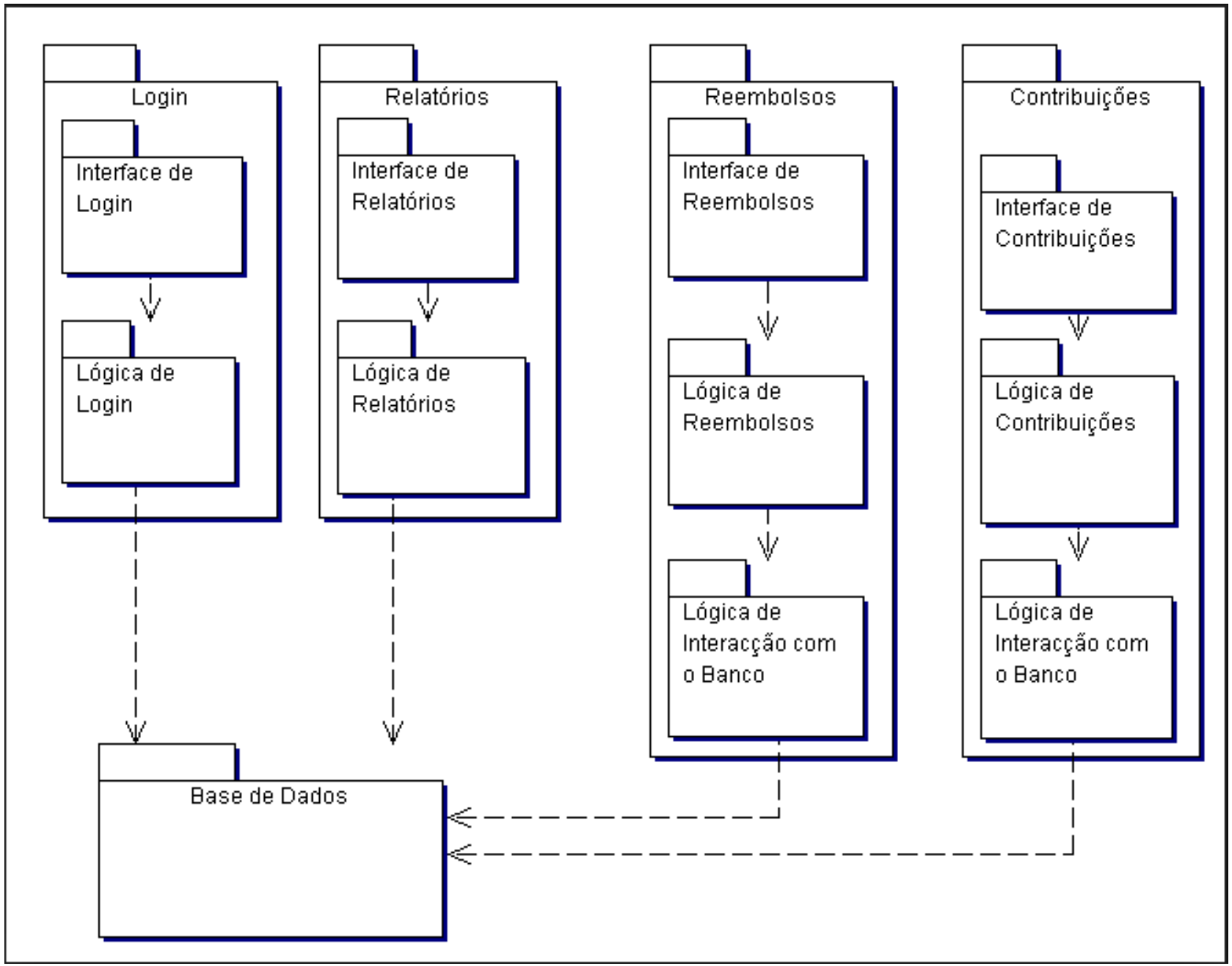
Exemplo:

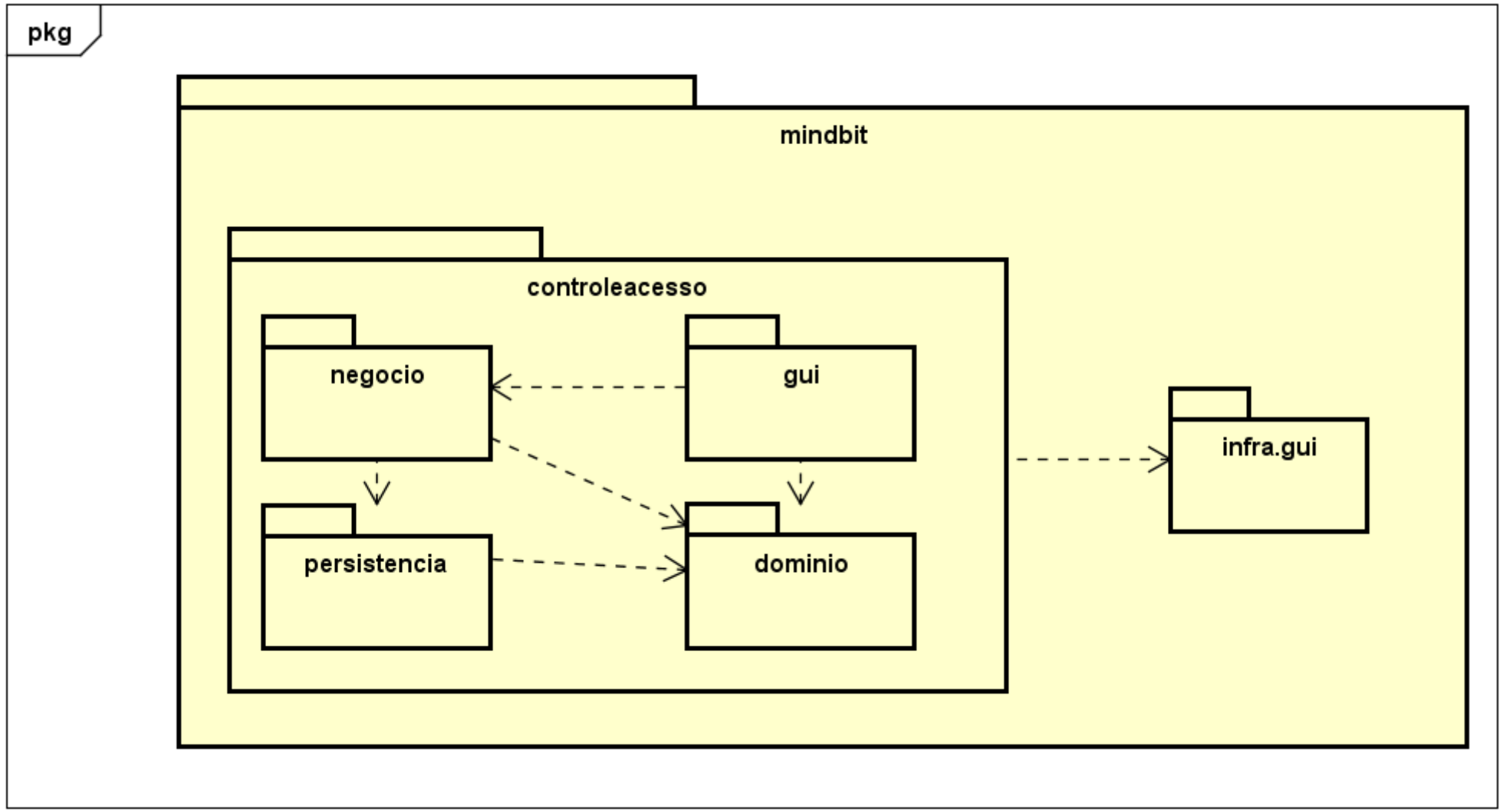
Exemplo

- Sistema segundo um modelo 3-camadas básicas
  - Elementos agrupados de acordo com a suas responsabilidades
  - Apresentação → responsável por exibir aos usuários as informações do sistema
  - Negócio → responsável pela implementação das regras de negócio
  - Persistência → responsável por armazenar e recuperar os dados de algum armazenamento persistente




Avaliação
Prova
Prova Discursiva
Prova Objetiva
Recurso
Vagas
Cargo
Localidade


Login
Interface de Login
Lógica de Login
Relatórios
Interface de Relatórios
Lógica de Relatórios
Reembolsos
Interface de Reembolsos
Lógica de Reembolsos
Lógica de Interacção com o Banco
Contribuições
Interface de Contribuições
Lógica de Contribuições
Lógica de Interacção com o Banco
Base de Dados


pkg
mindbit
controleacesso
negocio
gui
persistencia
dominio
infra.gui

Dicas e Sugestões

- Um pacote bem estruturado é:
  - Coeso → elementos claramente relacionados
  - Fracamente acoplado → exportando apenas os elementos que os outros pacotes precisam enxergar
  - Não contém muitos aninhamentos
  - Tem um conjunto equilibrado de conteúdos

Dicas e Sugestões

- Ao definir um pacote na UML:
  - Use a forma simples de ícone (só em último caso revele o seu conteúdo)
  - Ao revelar o conteúdo do pacote, mostre apenas os elementos necessários para a compreensão do significado

Exercício

- Usando uma ferramenta de sua escolha para a criação de UML (pode ser o próprio portal draw.io);
- Crie uma estrutura de pacotes da seguinte forma, atendendo ao MVC:
  - br.unipac.arquitetura.controller
  - br.unipac.arquitetura.service
  - br.unipac.arquitetura.DAO
  - br.unipac.arquitetura.domain
- Em cada um desses pacotes deverá ter uma classe com os nomes:
  - clienteController;
  - clienteService;
  - clienteDAO;
  - Cliente (pacote domain).
- Seguindo esse mesmo critério crie também classes para:
  - fornecedor;
  - cidade;
  - estado;
  - contasPagar;
  - contasReceber.

Dica: pesquise por "Controller Service DAO" para entender esse modelo

- Não é necessário nenhum atributo ou método nas classes;
- Pode ser feito em dupla

Modelagem Estrutural

Diagrama de Componentes

Diagrama de Componentes

- Componente é a parte lógica e substituível de um sistema ao qual se adapta e fornece a realização de um conjunto de interfaces

- Definem abstrações com interfaces bem definidas

- Interface é uma coleção de operações que especifica um serviço fornecido por ou solicitado de uma classe ou componente

Diagrama de Componentes

- Uma porta é um ponto específico de um componente que aceita mensagens
- A estrutura interna é a implementação de um componente por meio de um conjunto de partes que são conectadas de uma determinada maneira
- Uma parte é a especificação de um papel que compõe a implementação de um componente
- Um conector é um relacionamento de comunicação entre duas partes ou portas no contexto de um componente

Formas mais comuns – Diagrama de Componentes

Símbolo de componente

Entidade necessária para executar uma função de estereótipo.

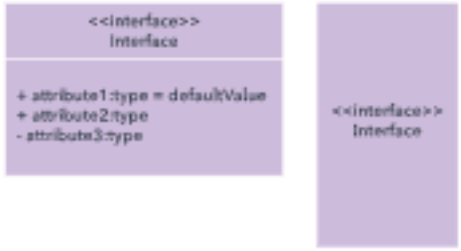
O componente fornece e consome comportamento em interfaces ou por meio de outros componentes.

Formas mais comuns – Diagrama de Componentes

Símbolo de nó

Representa objetos de hardware ou software, de nível superior aos componentes.

Formas mais comuns – Diagrama de Componentes



Símbolo de interface

Mostra entradas ou materiais que o componente recebe ou fornece

Formas mais comuns – Diagrama de Componentes

Especifica um ponto de interação separado entre o componente e o ambiente. O símbolo da porta é um pequeno quadrado.

Formas mais comuns – Diagrama de Componentes

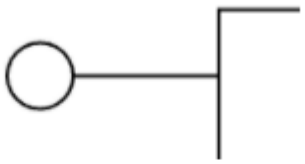
Símbolo de nota

Com ele, os desenvolvedores fixam uma meta-análise no diagrama de componentes.

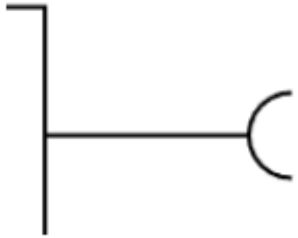
Formas mais comuns – Diagrama de Componentes

Mostra que uma parte do sistema depende de outra. A dependência é representada por linhas tracejadas que vinculam um componente (ou elemento) a outro.

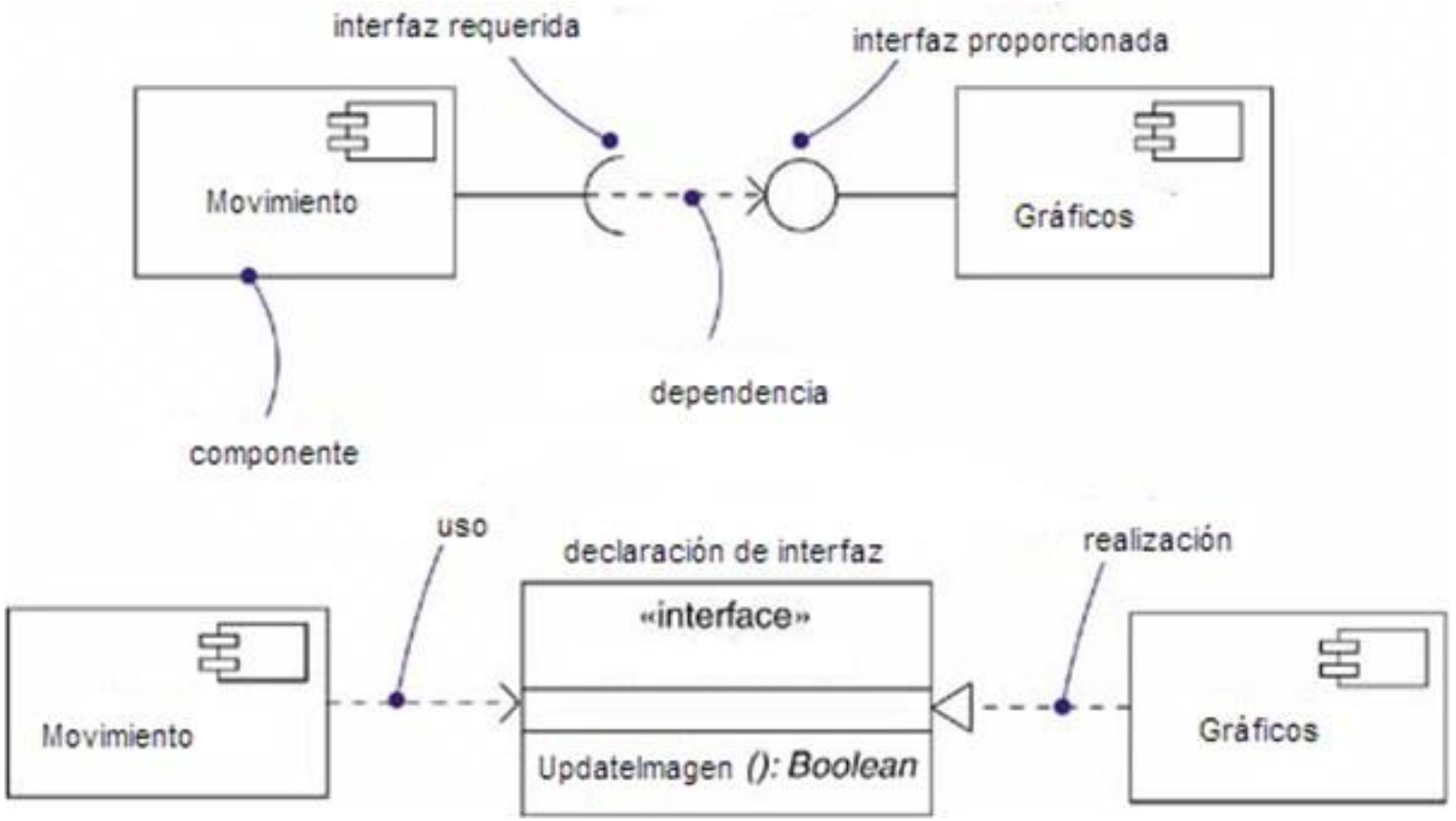
Formas mais comuns – Diagrama de Componentes

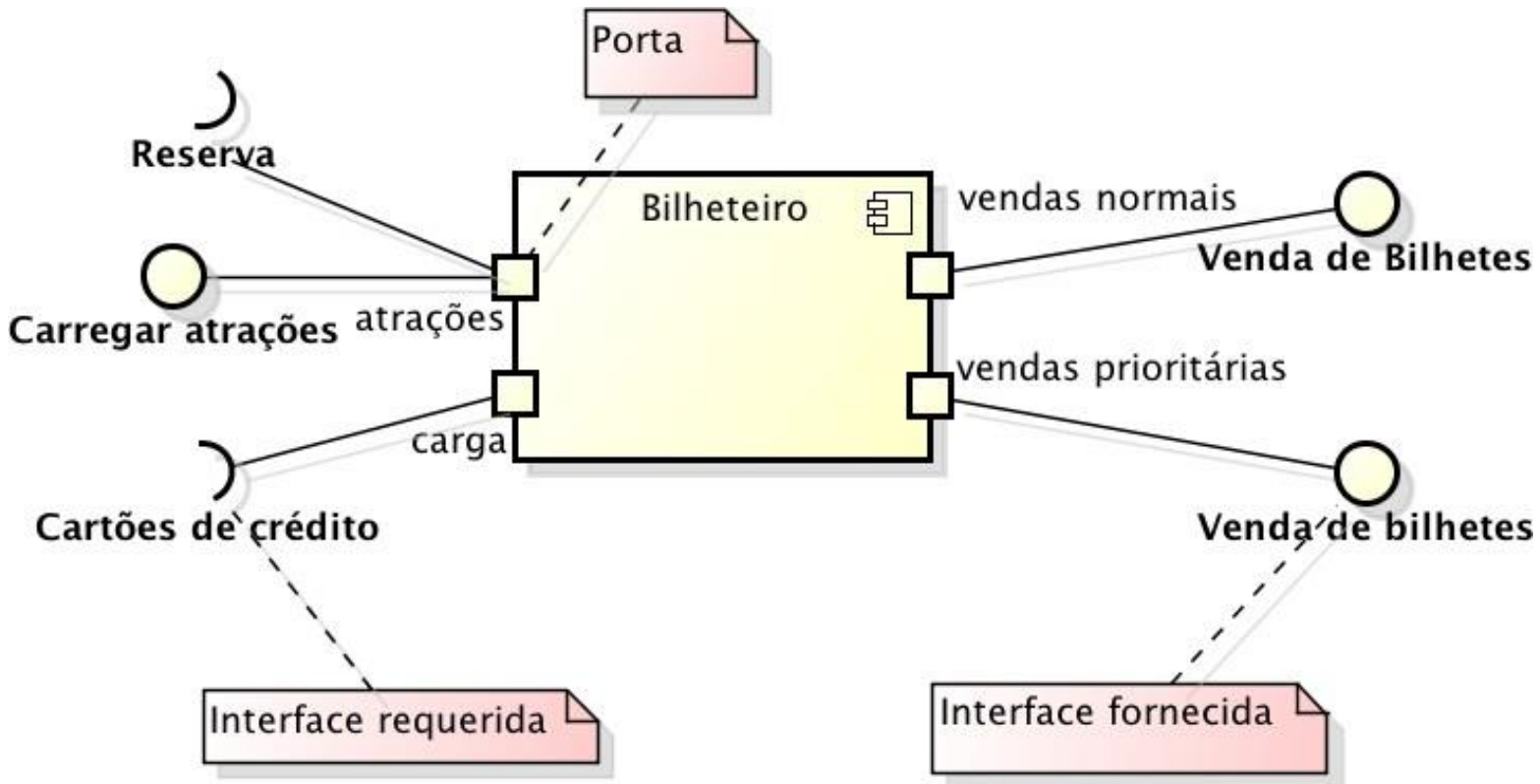


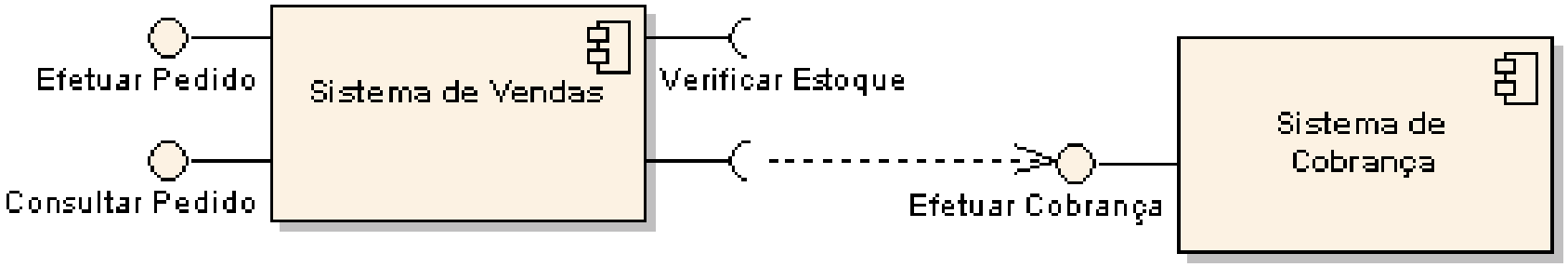
Interface fornecida: uma linha reta a partir da caixa de componentes e com um círculo anexado. Esse símbolo representa a interface na qual um componente produz informações usadas pela interface necessária de outro componente.

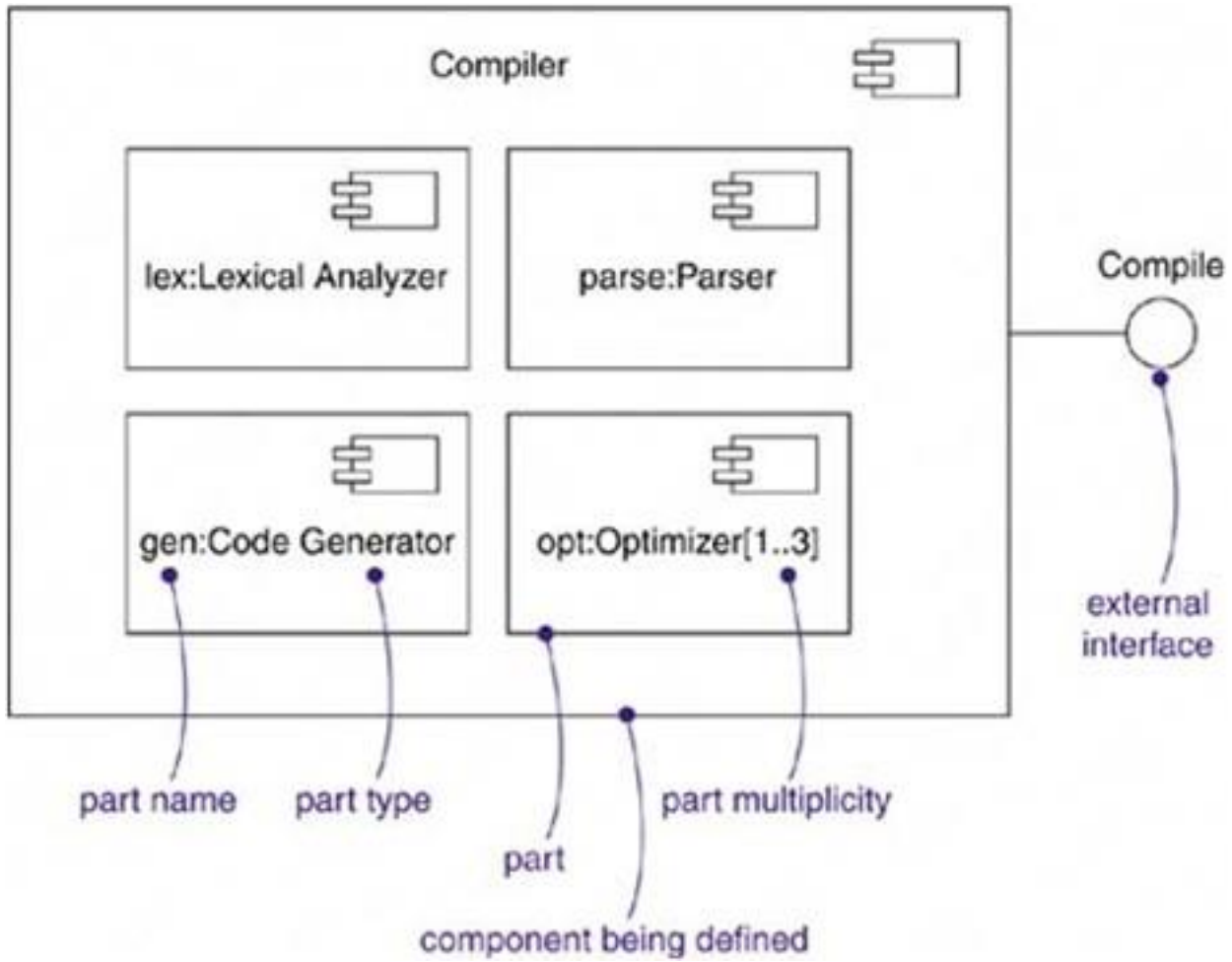


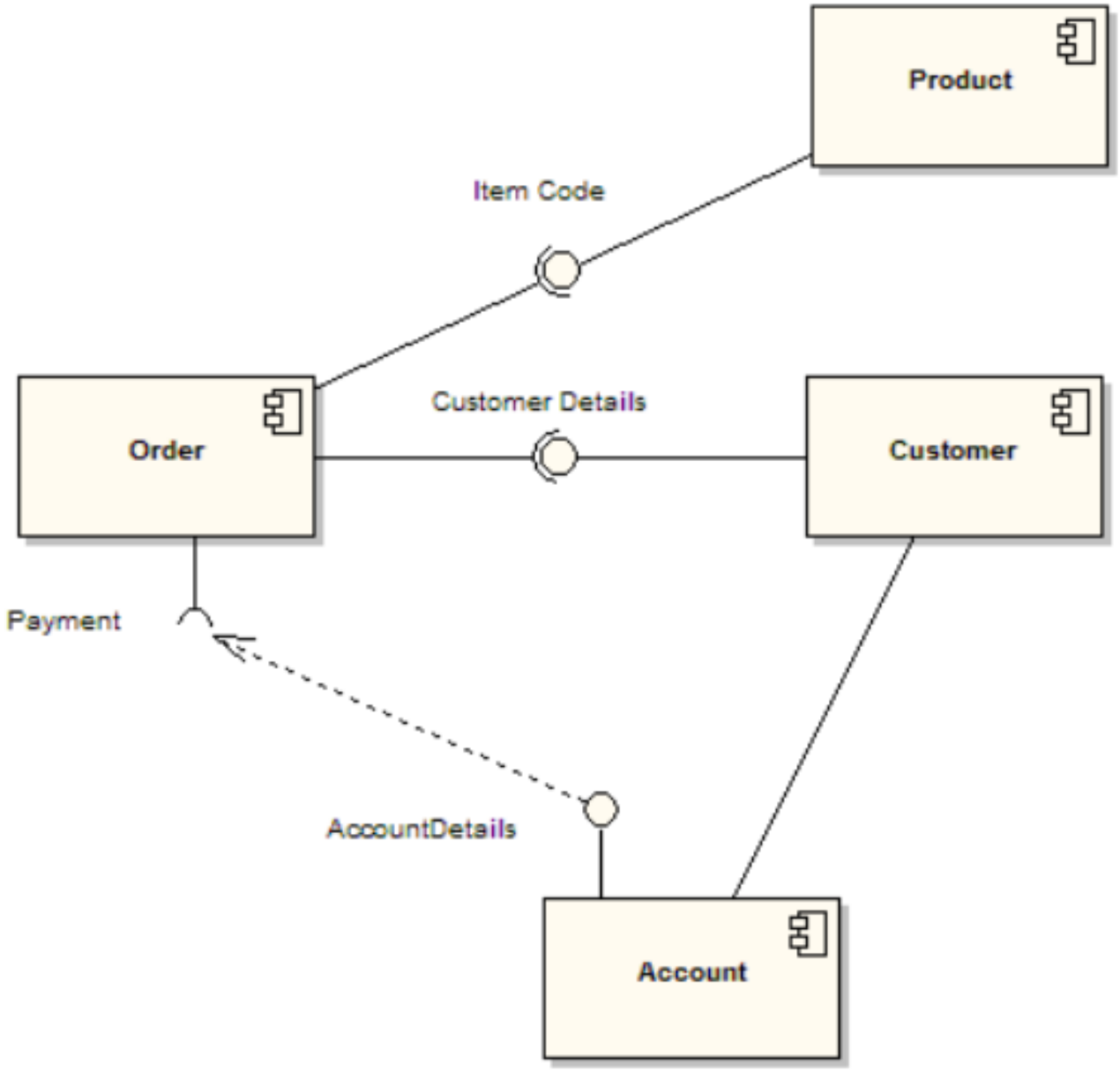
Interface necessária: uma linha reta da caixa de componentes e com um semicírculo anexado (também representado como uma seta tracejada com uma seta aberta). Esse símbolo representa a interface na qual um componente requer informações para executar a função correta.

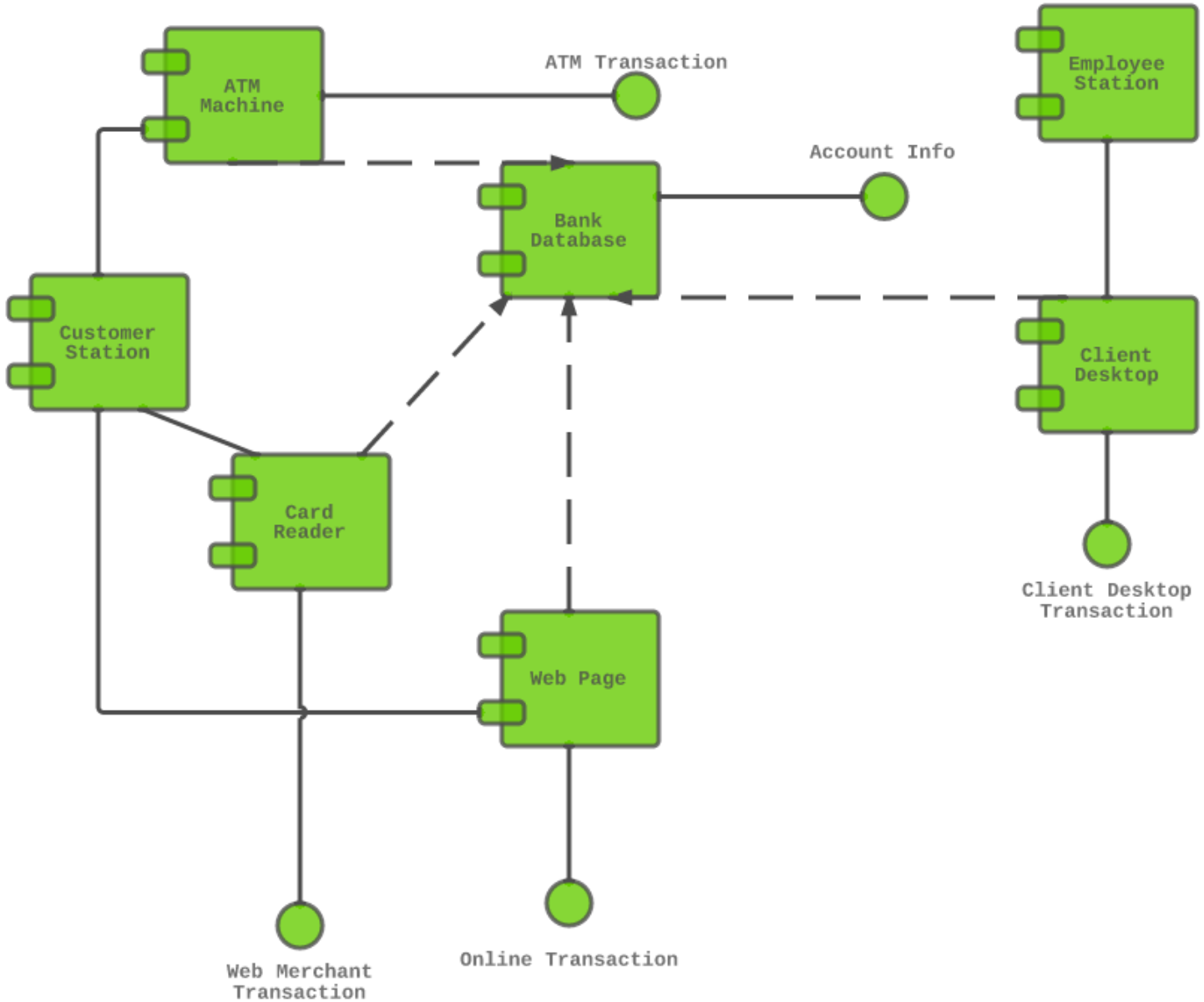

interfaz requerida
interfaz proporcionada
Movimiento
Gráficos
componente
dependencia
uso
declaración de interfaz
realización
«interface»
Movimiento
UpdateImagen (): Boolean
Gráficos


Porta
Reserva
Bilheteiro
vendas normais
Venda de Bilhetes
Carregar atrações
atrações
vendas prioritárias
carga
Cartões de crédito
Venda de bilhetes
Interface requerida
Interface fornecida


Efetuar Pedido
Consultar Pedido
Sistema de Vendas
Verificar Estoque
Efetuar Cobrança
Sistema de Cobrança


Compiler
lex:Lexical Analyzer
parse:Parser
gen:Code Generator
opt:Optimizer[1..3]
Compile
external interface
part name
part type
part
part multiplicity
component being defined

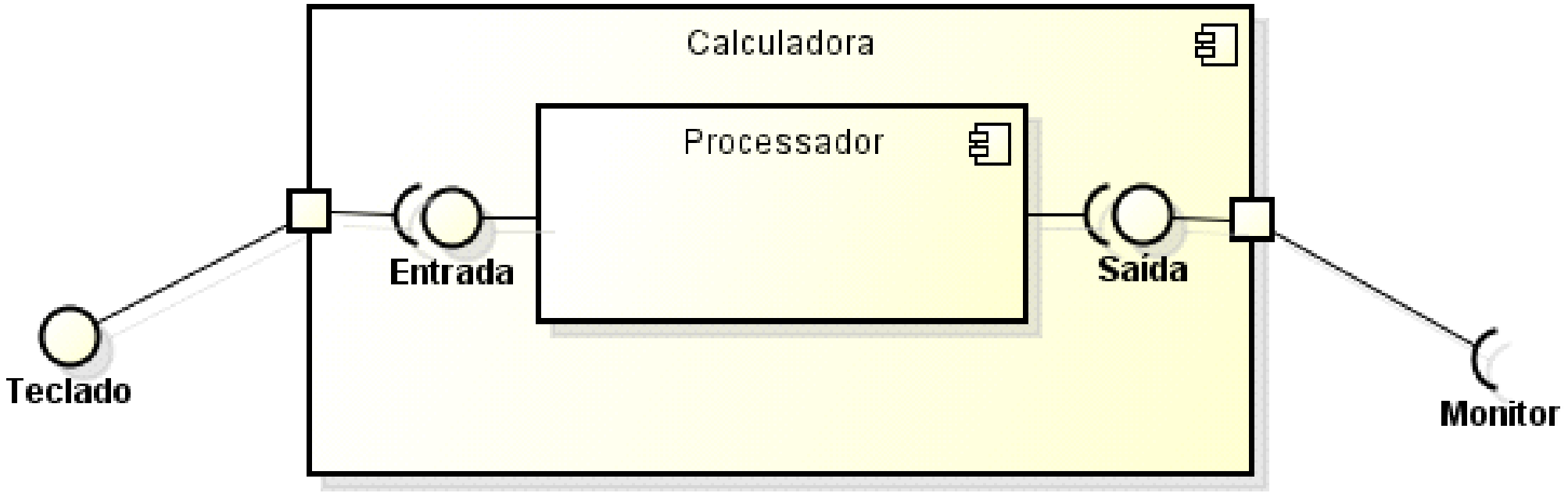

Product
Item Code
Order
Customer Details
Customer
Payment
AccountDetails
Account


ATM Machine
ATM Transaction
Employee Station
Account Info
Bank Database
Customer Station
Client Desktop
Card Reader
Client Desktop Transaction
Web Page
Web Merchant Transaction
Online Transaction

Dúvidas?

Prática durante a Aula

Precisamos criar um diagrama de componentes para demonstrar o funcionamento de uma calculadora, que terá a entrada dos dados pelo teclado, processará as operações e exibirá o resultado em um monitor.


Calculadora
Processador
Entrada
Saida
Teclado
Monitor

Fonte dos Slides

- https://slideplayer.com.br/slide/335723/
- https://www.lucidchart.com/pages/pt/diagrama-de-pacotes-uml
- https://www.lucidchart.com/pages/pt/diagrama-de-componentes-uml
- https://micreiros.com/diagramas-de-componentes/
- http://www.sparxsystems.com.ar/resources/tutorial/uml2_componentdiagram.php
- https://sites.google.com/site/appsmartcart/home/apresentacao/diagramas/diagrama-de-pacotes
- https://micreiros.com/diagrama-de-pacotes/
- https://sites.google.com/site/mindbitufrpe/diagramas/diagrama-de-pacotes