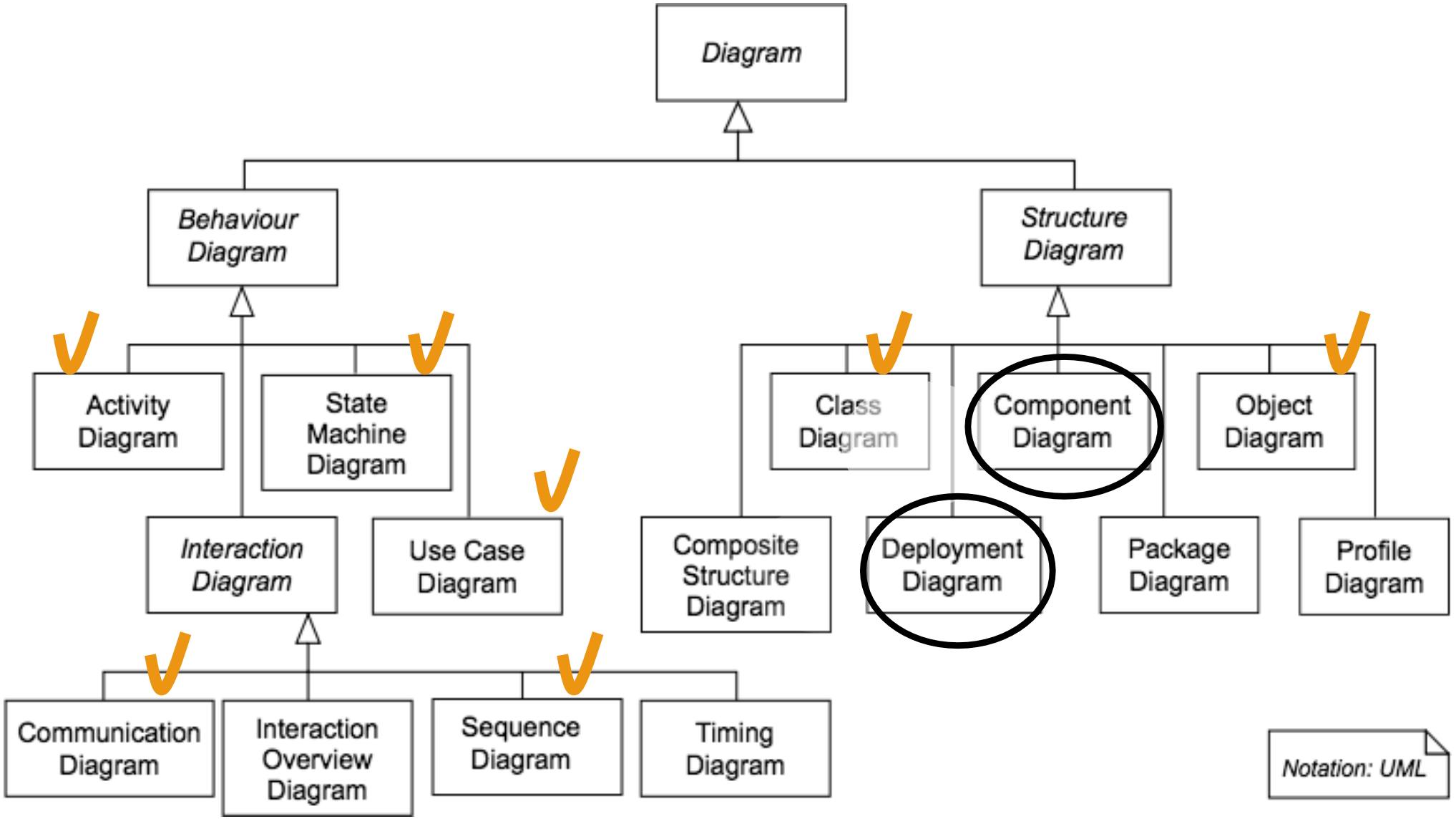
DIAGRAMAS ESTRUTURAIS

1. Diagrama de componentes
2. Diagrama de implantação (*deployment*)

Professores Andreia Malucelli, Edson Scalabrin e Marco Paludo

DIAGRAMAS ESTRUTURAIS



COMPONENTE E IMPLANTAÇÃO

A arquitetura física deve responder às seguintes questões:

- **Quais computadores** e outros dispositivos de hardware estão envolvidos no **processamento do sistema** e como estão **conectados entre** si ?
- Em quais **computadores as classes** e seus respectivos objetos **estão fisicamente conectados** ?
- Quais são as **dependências entre diferentes arquivos** de código?
- Se um arquivo específico é alterado, quais outros arquivos devem ser recompilados ?

DIAGRAMA DE COMPONENTES

COMPONENTES

Componentes são um conceito ligeiramente confuso na UML, porque tanto classes quanto componentes podem ser usados para modelar a mesma coisa. Por exemplo, citando Rumbaugh:

- **A distinção entre uma classe estruturada e um componente é algo vago e mais uma questão de intenção do que de semântica rígida. [RJB04]**

E para citar a especificação UML [OMG03b]:

Um componente representa uma parte modular de um sistema que encapsula seu conteúdo e cuja manifestação é substituível dentro de seu ambiente. Um componente define seu comportamento em termos de interfaces fornecidas e requeridas. Como tal, um componente serve como um tipo, cuja conformidade é definida por essas interfaces fornecidas e requeridas.

Larman, Craig.

DIAGRAMA DE COMPONENTES

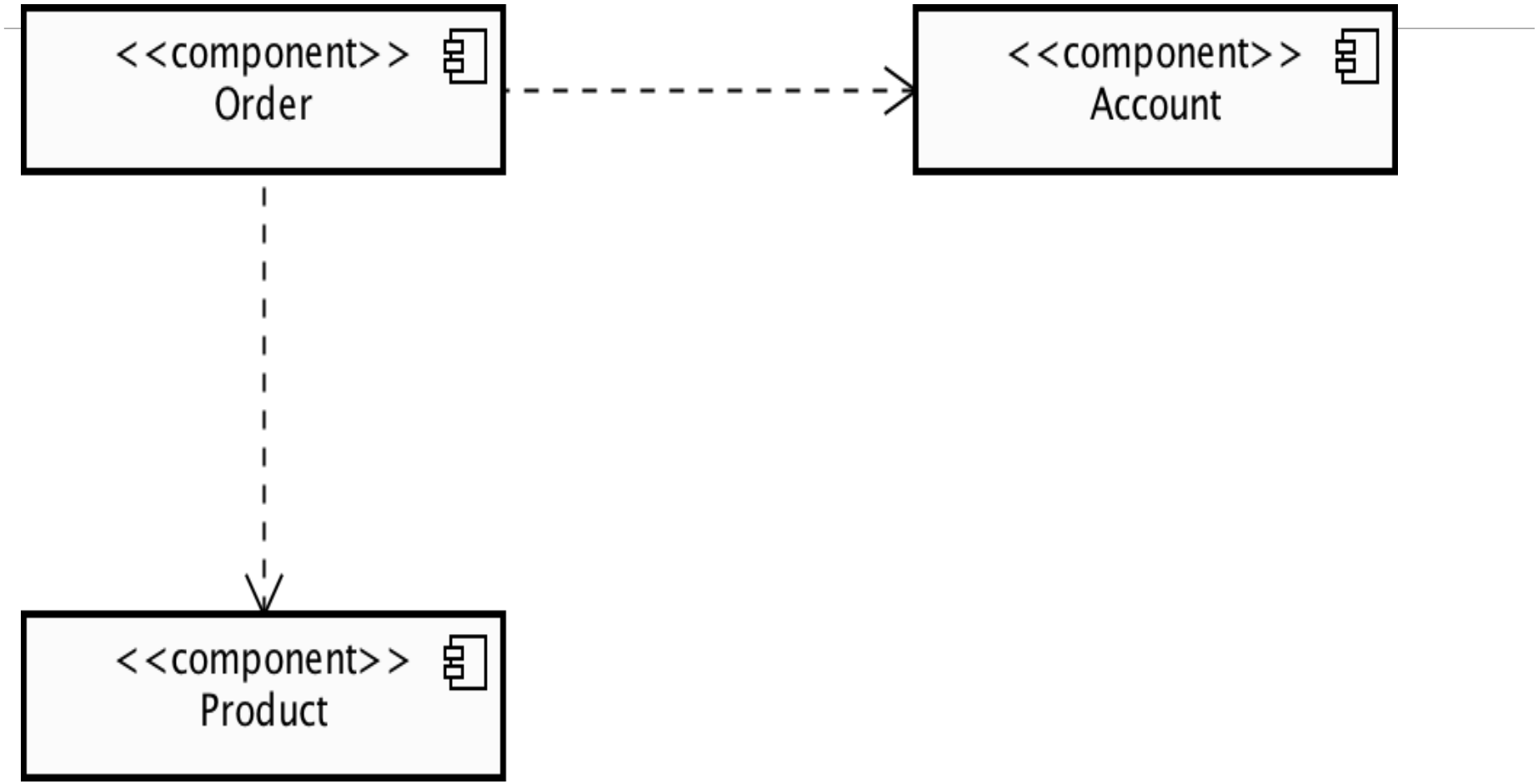
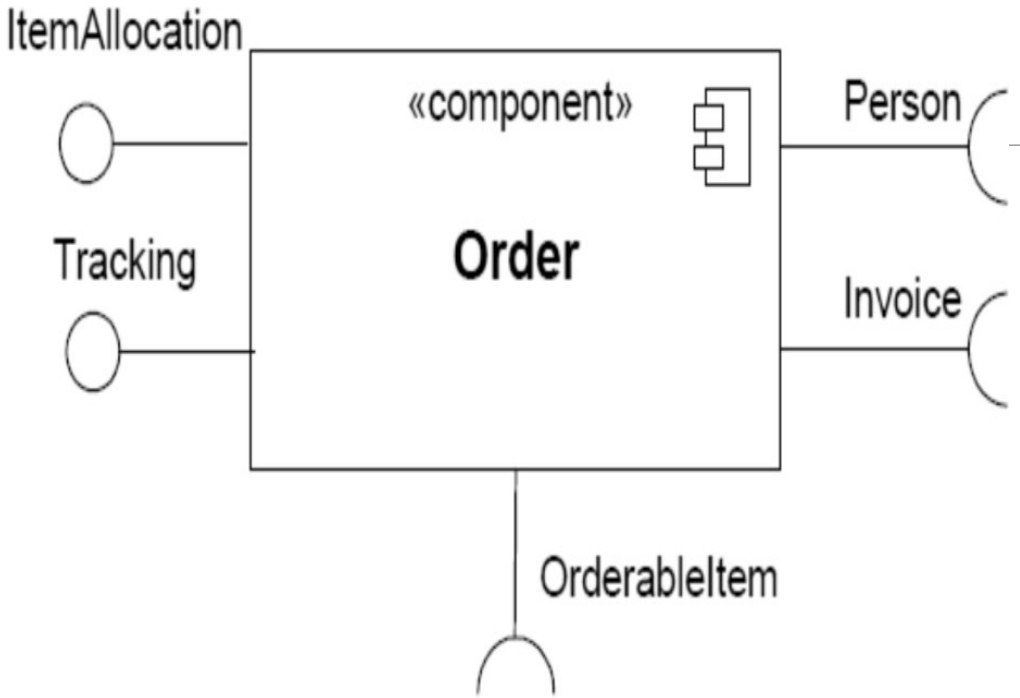
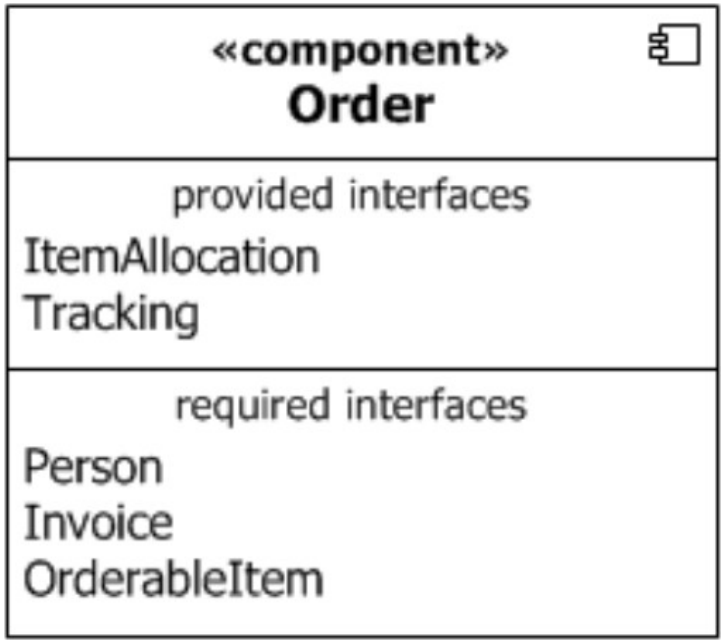


DIAGRAMA DE COMPONENTES





Duas interfaces fornecidas e três requeridas

DIAGRAMA DE COMPONENTES

Apresenta uma visão estática de como o sistema será implementado e quais os seus módulos de software, ou seja, os seus **componentes**.

Este diagrama está associado à linguagem de programação em que o projeto será implementado.

Procura identificar os arquivos que comporão o software em termos de módulos, bibliotecas, formulários, arquivos de help, documentos, etc., além de determinar como os componentes se relacionam entre si.

DIAGRAMA DE COMPONENTES

Pode ser utilizado para modelar componentes do código-fonte do sistema, do código-executável do software ou bancos de dados físicos.

Destaca a função de cada módulo para facilitar sua reutilização em outros sistemas.

Pode ser utilizado para auxiliar no processo de engenharia reversa, por meio da organização dos módulos do sistema e seus relacionamentos.

DIAGRAMA DE COMPONENTES

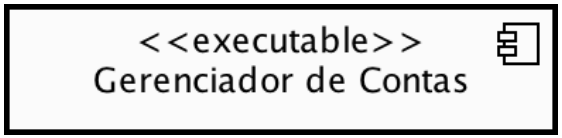
Exibe as organizações e dependências entre componentes com propósito de modelar a visão de implementação dos módulos de software executáveis com identidade e interface bem-definida de um sistema e seus relacionamentos.

Por exemplo, uma classe do modelo lógico pode mapear dois arquivos em uma implementação de C++: um arquivo de .h para a definição de classe e um arquivo de .cpp para as definições de função das classes membros.

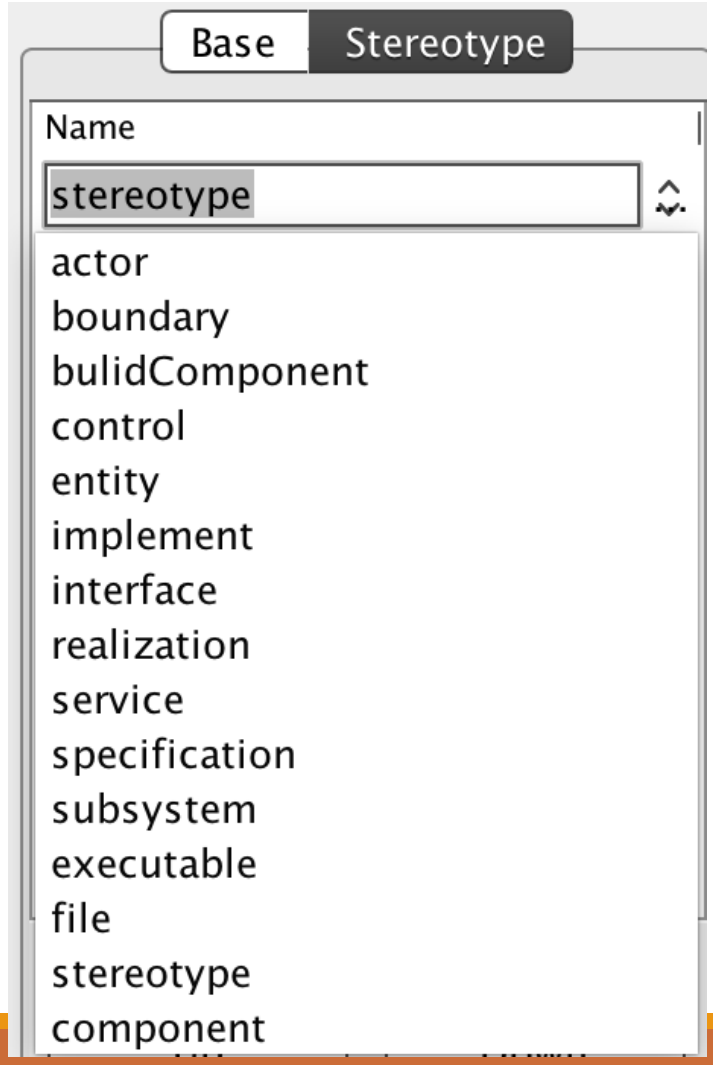
ESTEREÓTIPOS

Os estereótipos podem ser utilizados para destacar a função de um determinado componente. A UML prevê diversos estereótipos para o Diagrama de Componentes:

- Executável <<executable>>
- Biblioteca <<library>>
- Documento <<document>>
- Arquivo <<file>>



- Se houver necessidade podem ser criados estereótipos próprios.



ESTEREÓTIPOS

Executável: ***componente é um arquivo executável***, ou seja, um arquivo já compilado e link-editado em linguagem de máquina que pode executar uma série de instruções.

Biblioteca: ***refere-se a bibliotecas contendo funções que podem ser compartilhadas por diversos componentes*** executáveis, podendo estas serem fornecidas pela própria linguagem em que o sistema será desenvolvido ou serem criadas pelos próprios desenvolvedores do sistema.

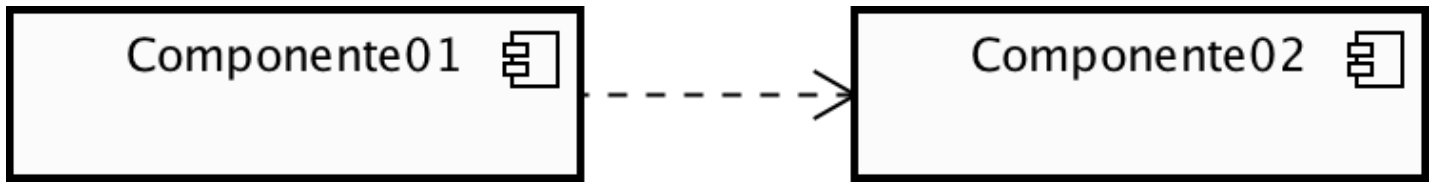
ESTEREÓTIPOS

Documento: **refere-se a arquivos-texto** utilizados por outros componentes do sistema, como arquivos de ajuda (help).

Arquivo: ***refere-se a qualquer outro arquivo*** que componha o sistema, como arquivos de código-fontes dos módulos do sistema.

DEPENDÊNCIA

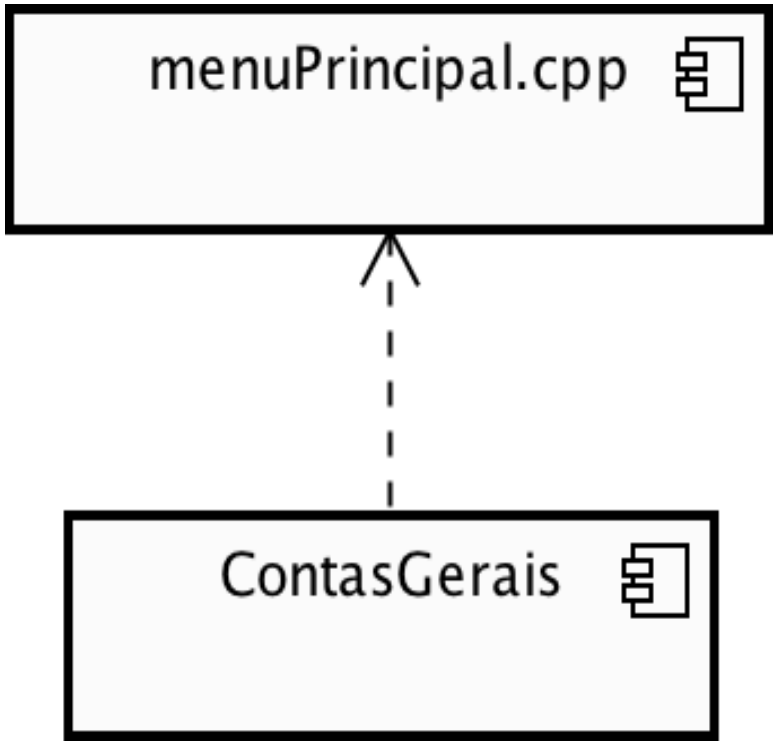
1. Um componente pode utilizar serviços ou depender de alguma outra forma de outros componentes do sistema. Isso é determinado pelo relacionamento de dependência.
2. Representado por uma reta tracejada contendo uma seta que indica que este componente depende de outro componente.



DEPENDÊNCIA

Exemplo de 2 módulos em C++.

Módulo de ContasGerais possui um certo grau de dependência com o módulo principal, já que será por meio deste que o módulo de Contas poderá ser utilizado.



DEPENDÊNCIA

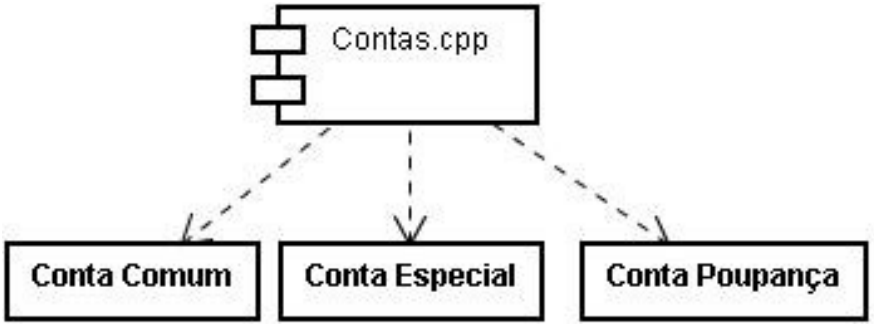
Também pode ser utilizado para mostrar as classes que estão sendo implementadas ou manipuladas por um componente.

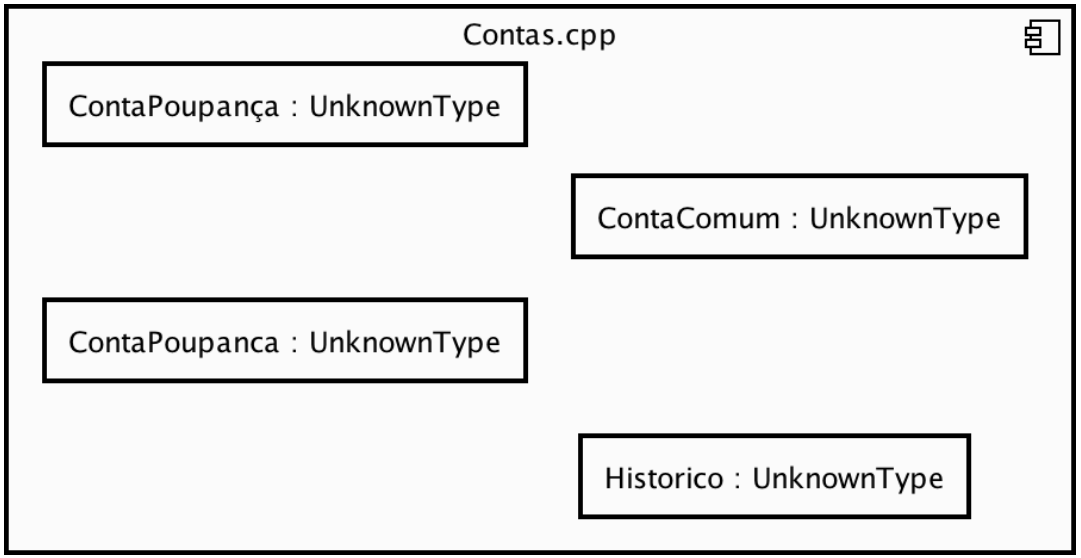
As classes são representadas por um retângulo.

Vamos a um exemplo

DEPENDÊNCIA

Exemplo onde o componente Contas.cpp manipula as classes Conta Comum, Conta Especial e Conta Poupança.





Representação de Interface e Porta

USANDO PACOTES EM UML, INTERFACE E PORTA

INTERFACE

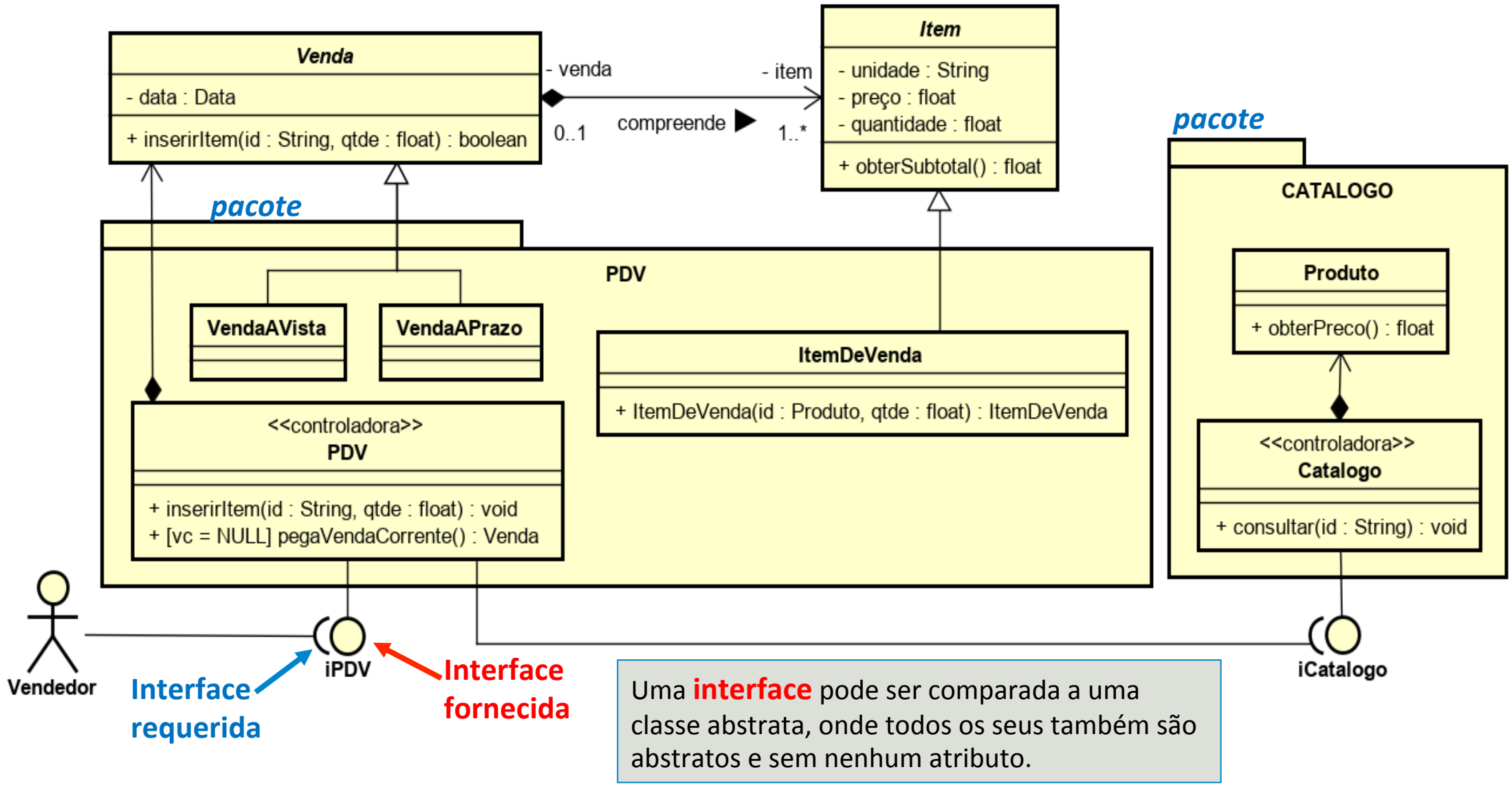
Representa um serviço realizado por uma Classe ou Componente.

Não possuem implementação ou qualquer especificação interna.

É representada por um ***pequeno círculo*** e pode possuir dois tipos de relacionamento com os componentes:

1. se um ***componente implementa*** alguma função da Interface, este relaciona-se com ela através de um relacionamento de realização (linha reta);

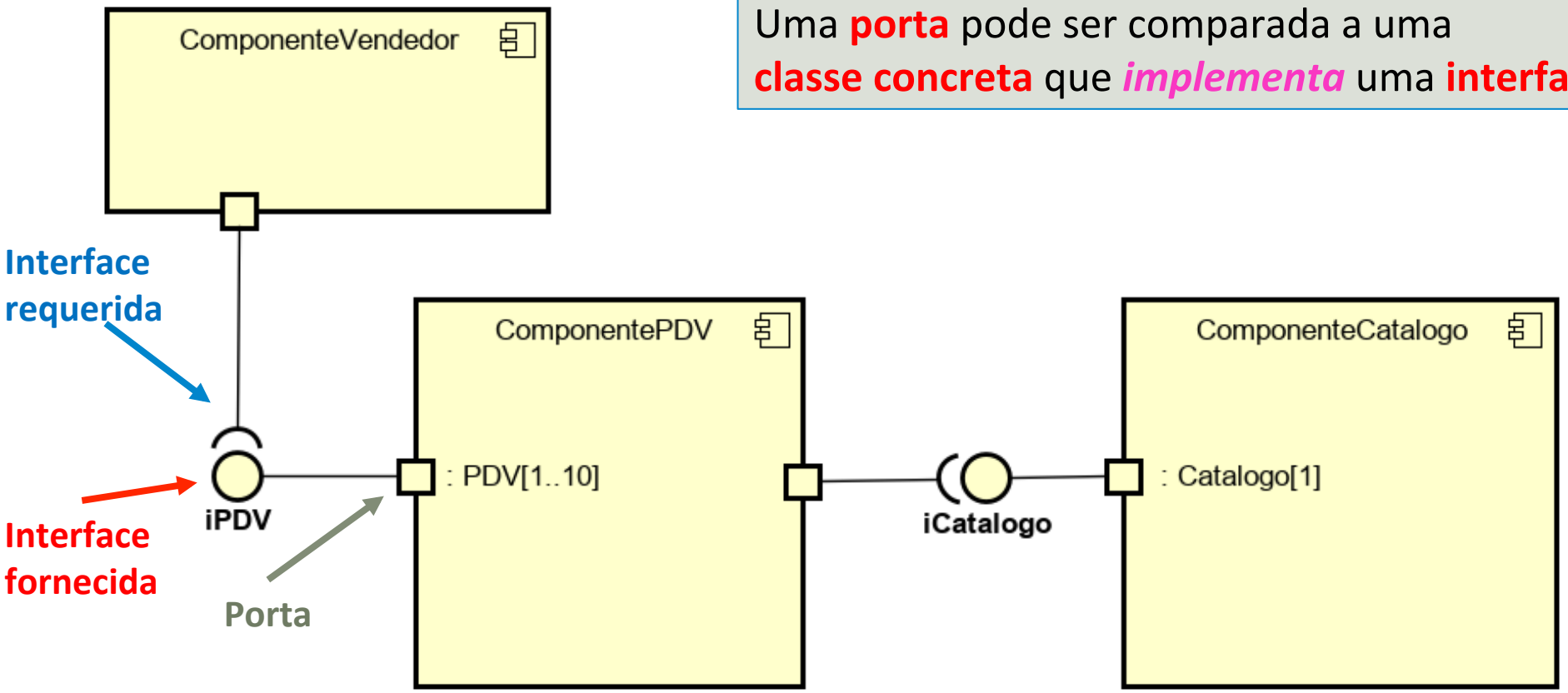
2. se um ***componente utiliza*** a interface, este se relaciona com ela através de um relacionamento de Dependência.



Uma **interface** pode ser comparada a uma classe abstrata, onde todos os seus também são abstratos e sem nenhum atributo.

Uma **interface** pode ser comparada a uma classe abstrata, onde todos os seus também são abstratos e sem nenhum atributo.

Uma **porta** pode ser comparada a uma **classe concreta** que ***implementa*** uma **interface**.



Um **Componente** é uma **unidade independente** que encapsula o **estado e o comportamento** de um número de **Classificadores**.

Figura A.2 é um diagrama de classes de um pacote P: as classes C1 e C2 são definidas no **namespace** do **pacote P.**

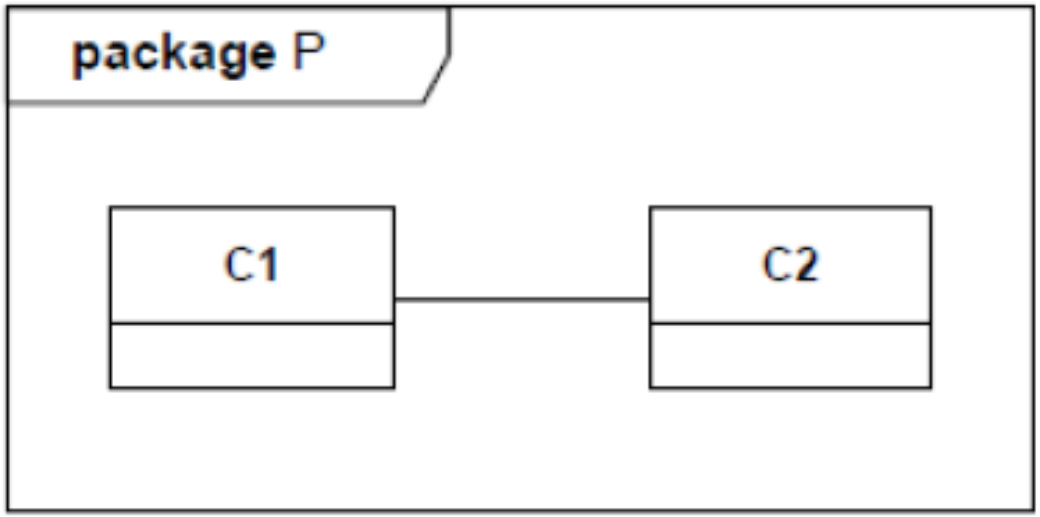


Figure A.2 Class diagram of package P

A Figura A.3 ilustra que um símbolo de pacote para o pacote P (em alguns pacotes CP) pode mostrar o mesmo conteúdo que o diagrama de classes para o pacote.

É um diagrama de pacotes para o pacote CP com símbolos gráficos que representam o fato de que o pacote CP contém um pacote P.

É um diagrama de classes para o pacote P.

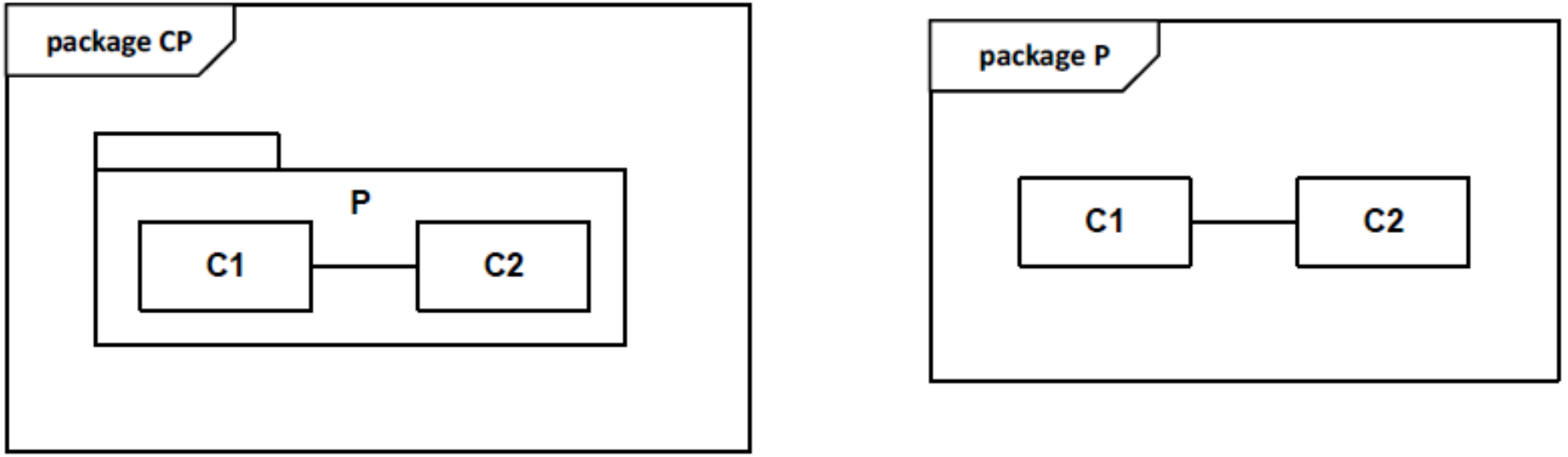


Figure A.3 Two diagrams of packages

É um diagrama de classes para o **pacote Cars**, com um símbolo de classe representando o fato de que o pacote Cars contém uma classe Car.

É um diagrama de estrutura composto para a **classe Carro**. O símbolo da classe em si não precisa conter a estrutura da classe em um compartimento; para modelos mais realistas, os símbolos das classes normalmente têm apenas os nomes das classes, enquanto que os diagramas de estrutura composta para as classes têm símbolos para as estruturas compostas.

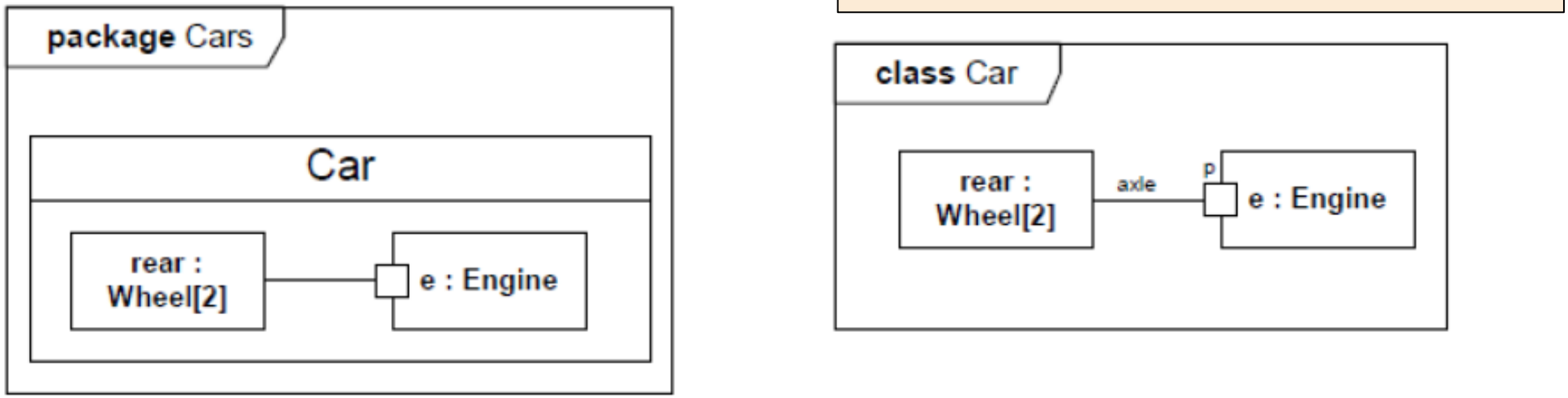
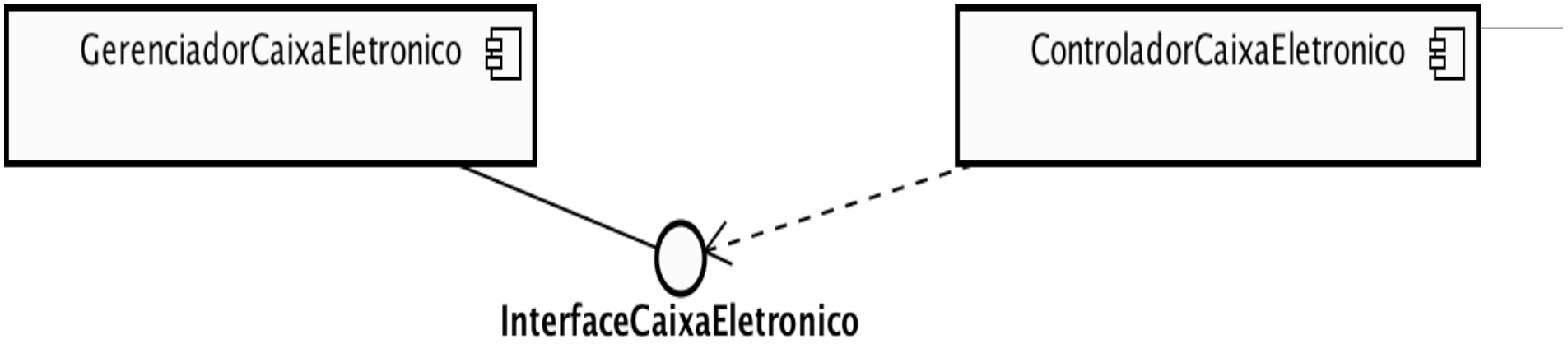


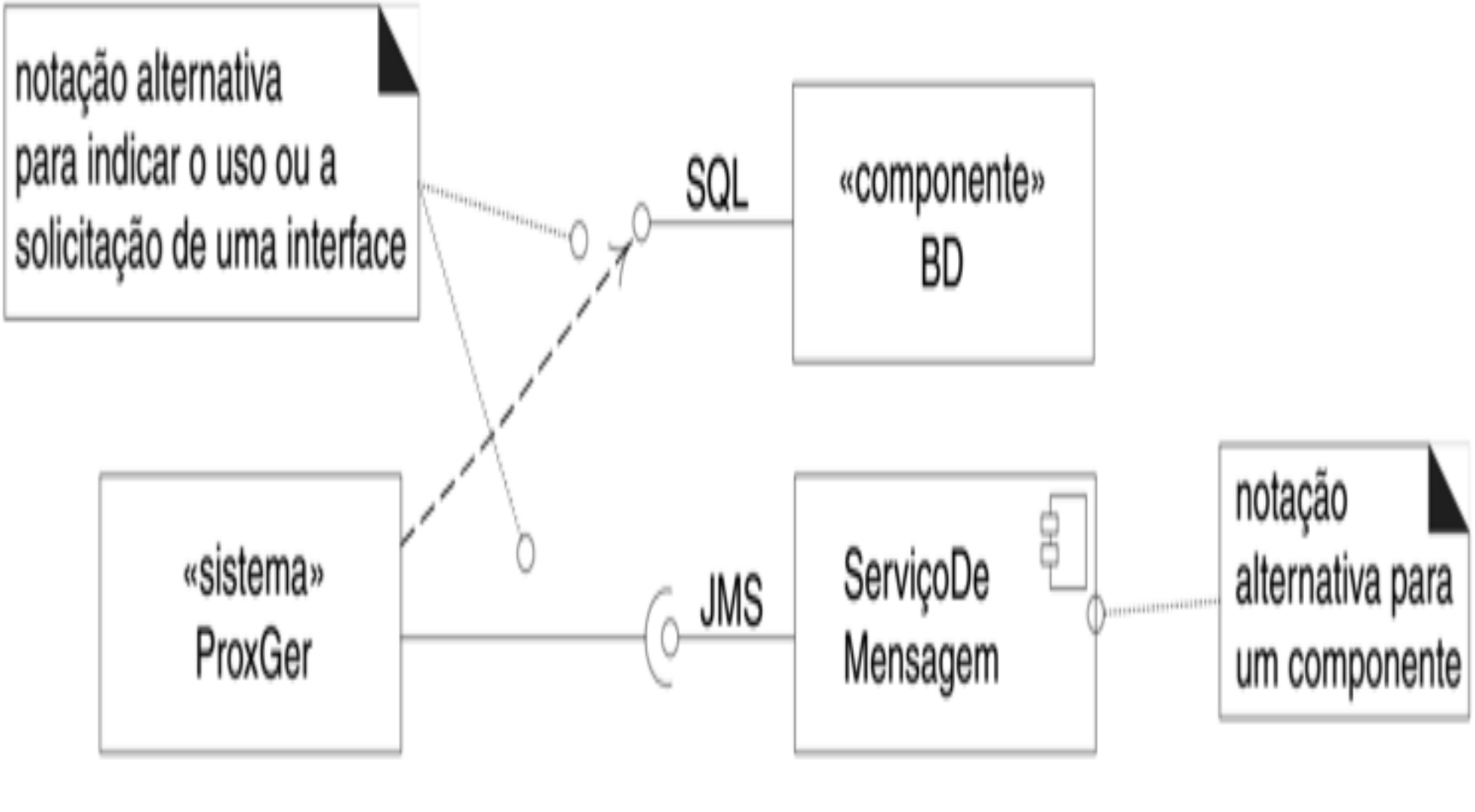
Figure A.4 A class diagram and a composite structure diagram

INTERFACE



O componente Gerenciador de Caixa Eletrônico implementa a Interface chamada Caixa Eletrônico e esta Interface é utilizada pelo componente Controlador do Caixa Eletrônico.

INTERFACE


notação alternativa
para indicar o uso ou a
solicitação de uma interface
SQL
«componente»
BD
«sistema»
ProxGer
JMS
ServiçoDe
Mensagem
notação
alternativa para
um componente

INTERFACE

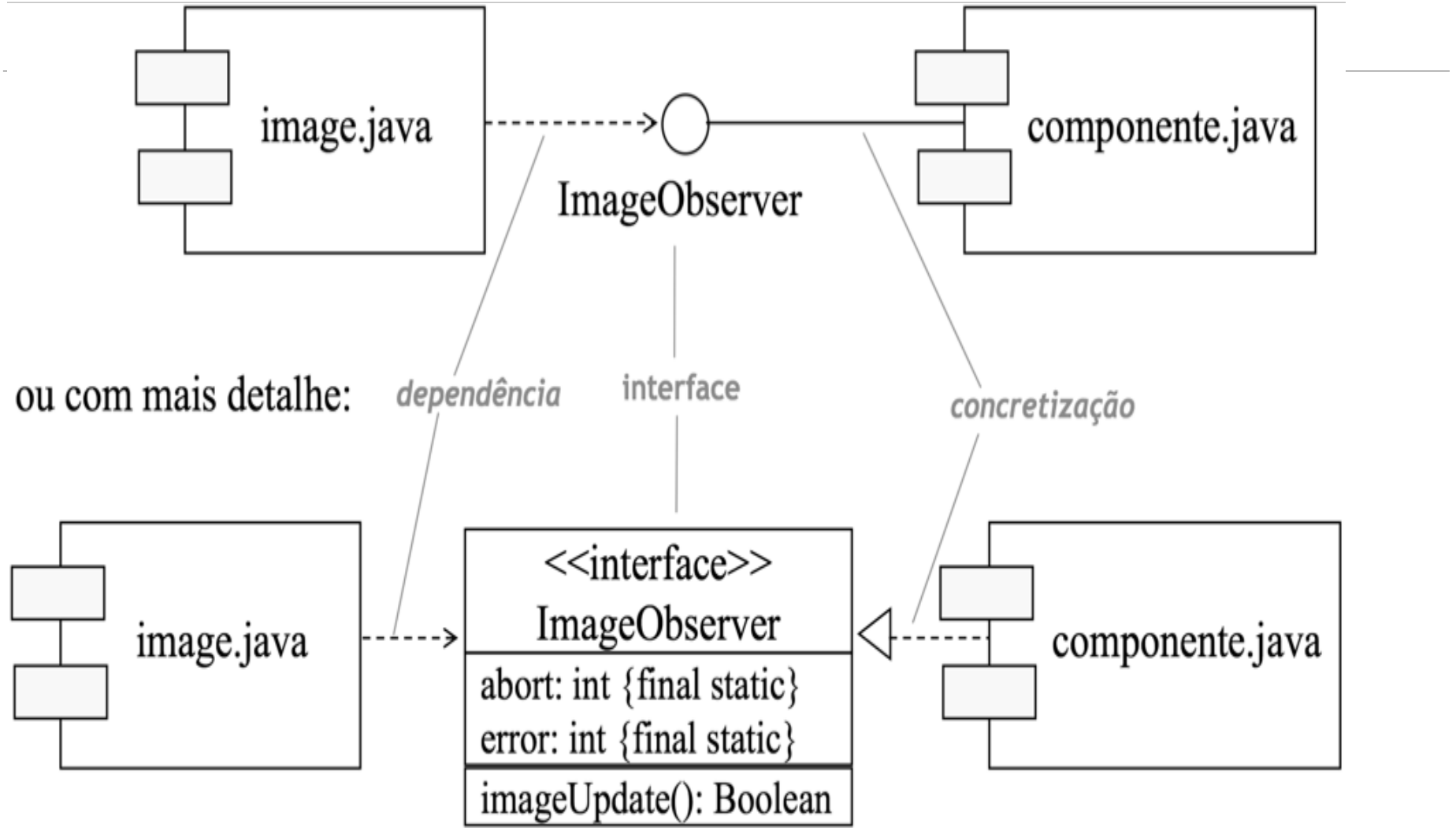
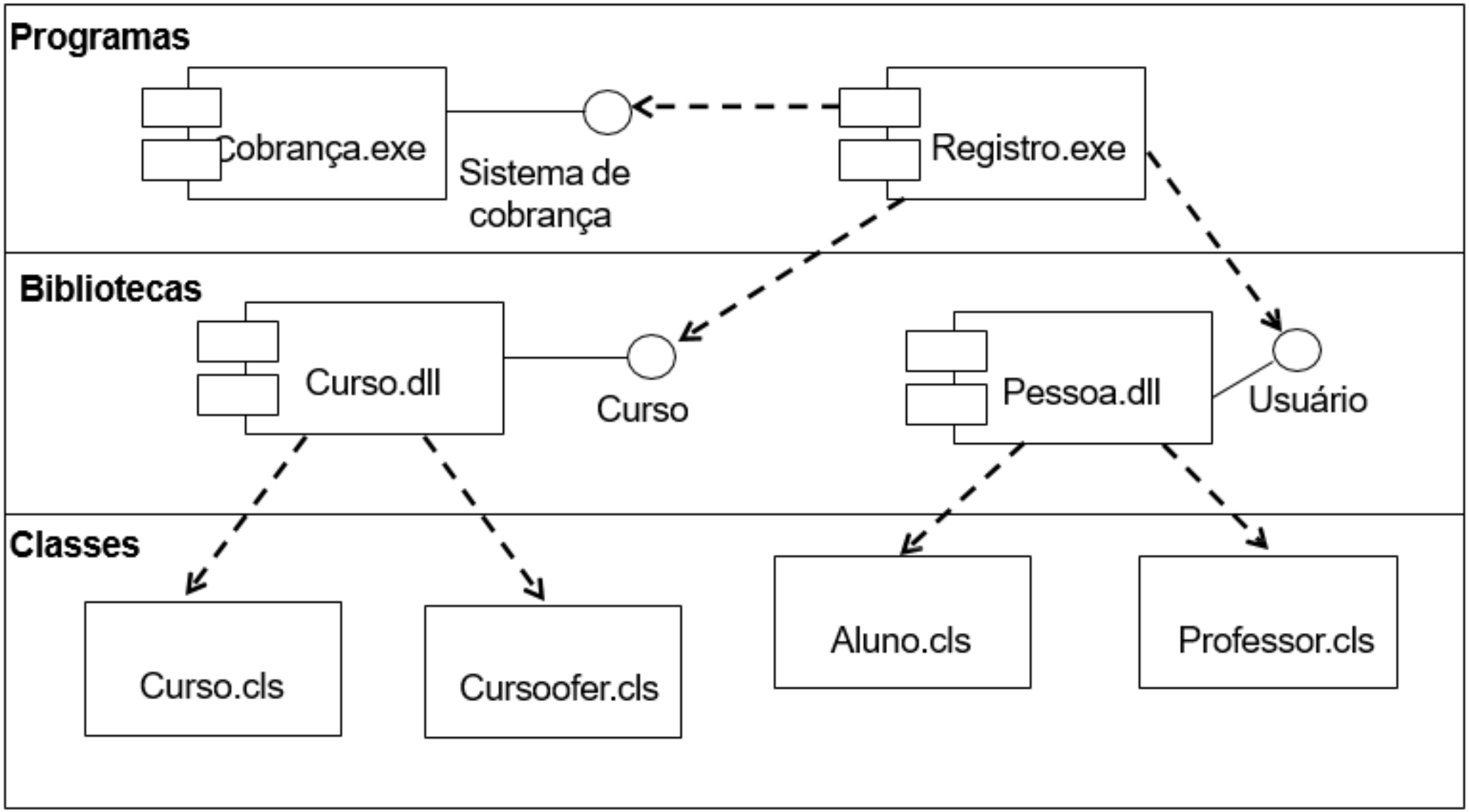


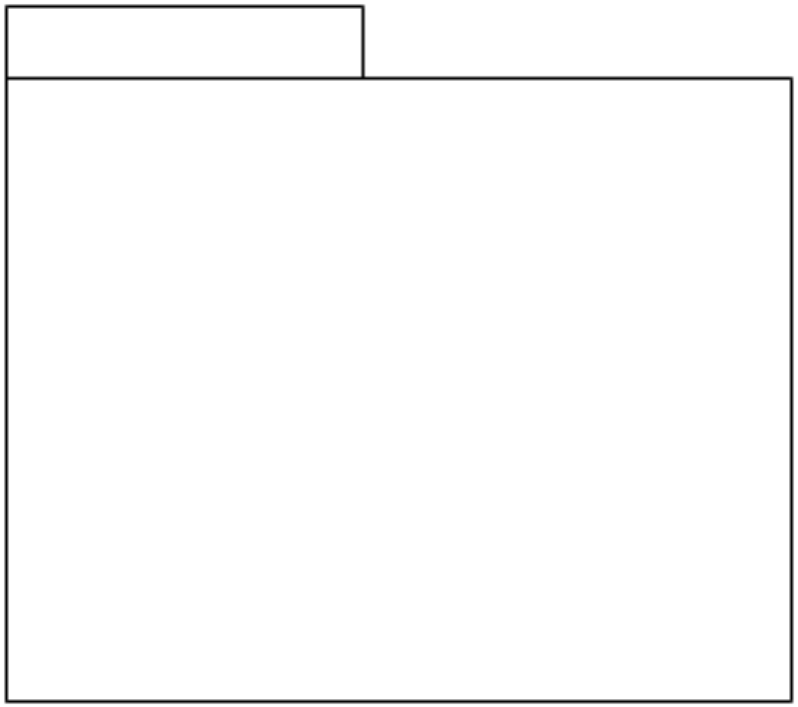
image.java
ImageObserver
componente.java
ou com mais detalhe:
dependência
interface
concretização
image.java
<<interface>>
ImageObserver
abort: int {final static}
error: int {final static}
imageUpdate(): Boolean
componente.java

DIAGRAMA DE COMPONENTES



PACOTES

Os sub-sistemas em que um sistema eventualmente se divide podem ser representados por meio da utilização de pacotes.



PACOTES

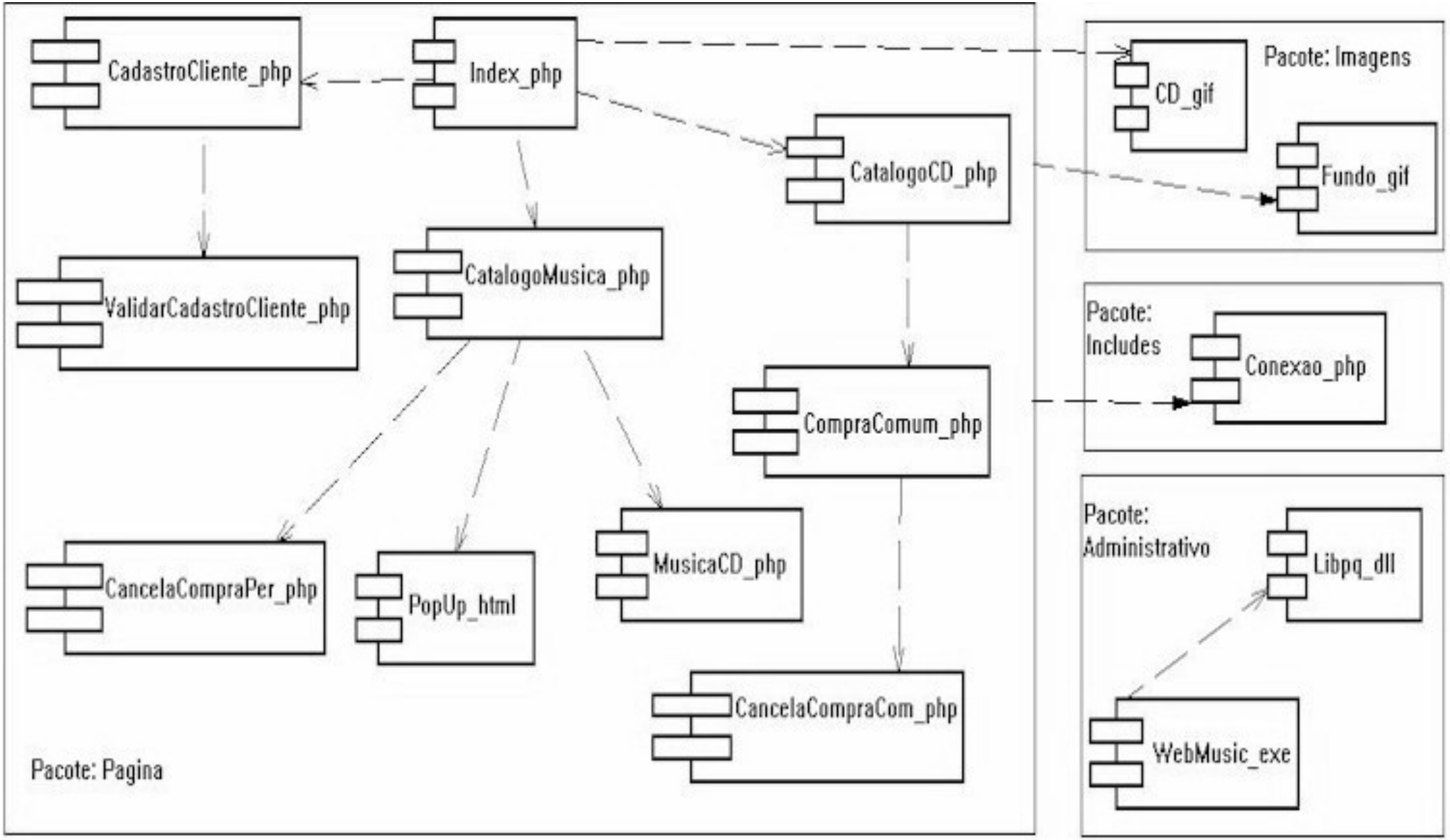
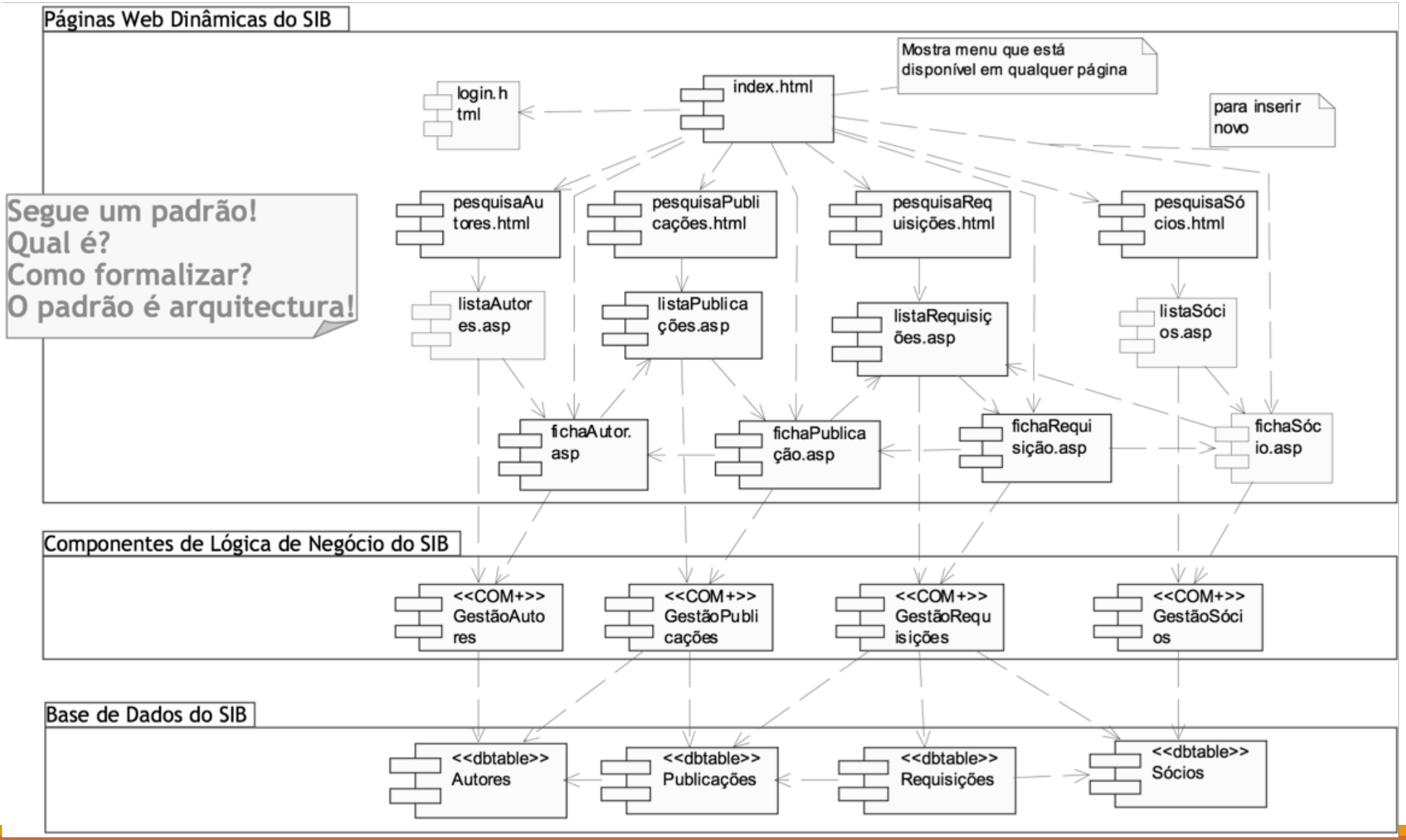

CadastroCliente_php
Index_php
CatalogoCD_php
CatalogoMusica_php
ValidarCadastroCliente_php
CompraComum_php
CancelaCompraPer_php
PopUp_html
MusicaCD_php
CancelaCompraCom_php
Pacote: Pagina
Pacote: Imagens
CD_gif
Fundo_gif
Pacote:
Includes
Conexao_php
Pacote:
Administrativo
Libpq_dll
WebMusic_exe

DIAGRAMA DE COMPONENTES EM PACOTES



DÚVIDAS?

DIAGRAMA DE IMPLANTAÇÃO (DEPLOYMENT)

DIAGRAMAS ESTRUTURAIS

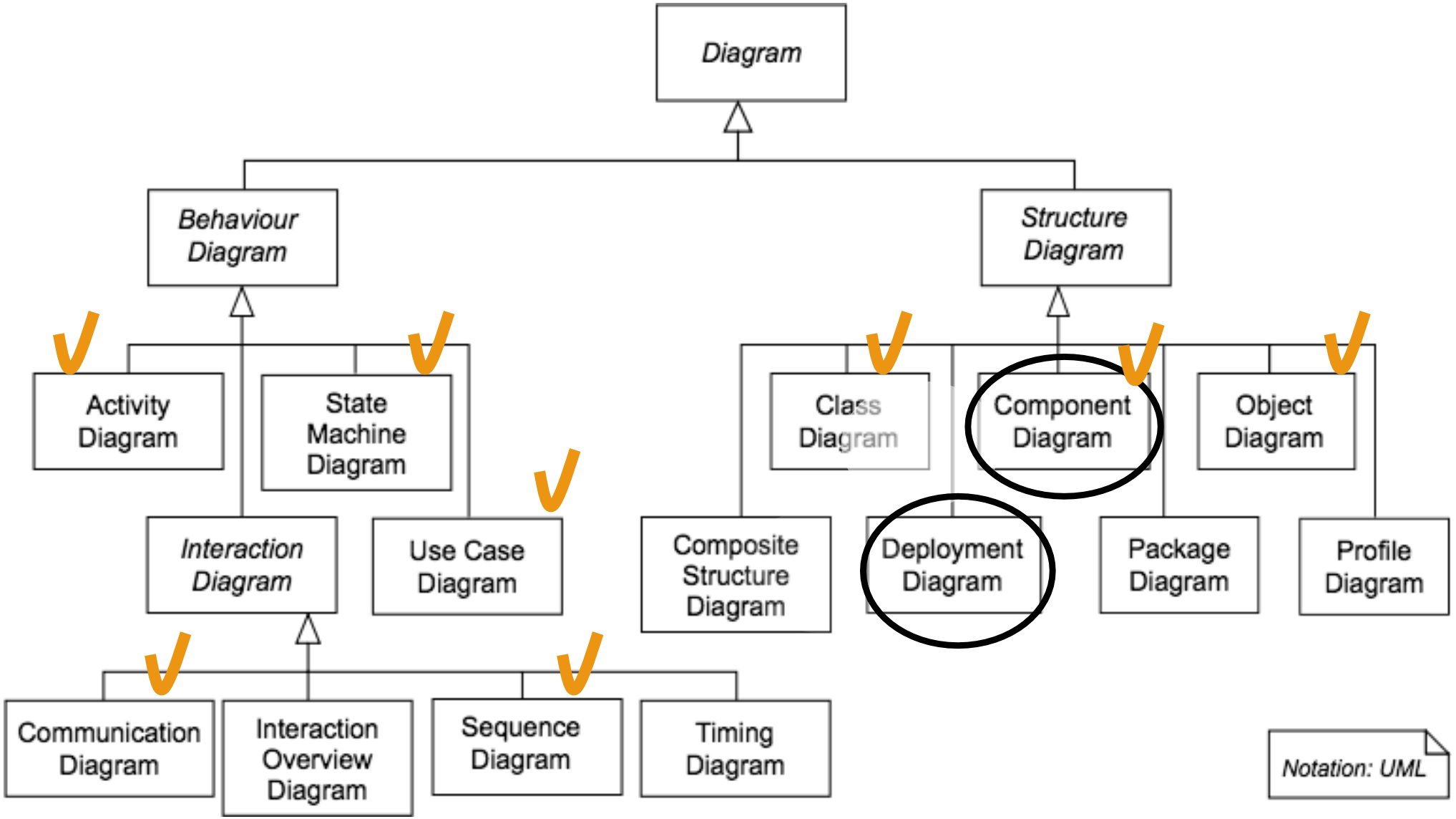


DIAGRAMA DE IMPLANTAÇÃO

Um ***diagrama de implantação*** mostra a atribuição de artefatos concretos de software (como arquivos executáveis) a ***nós computacionais*** (algo com serviços de processamento).

Mostra a implantação de elementos de software à **arquitetura física** e a **comunicação** (geralmente em uma rede) entre elementos físicos.

Diagramas de implantação são ***úteis para comunicar a arquitetura física*** ou de implantação.

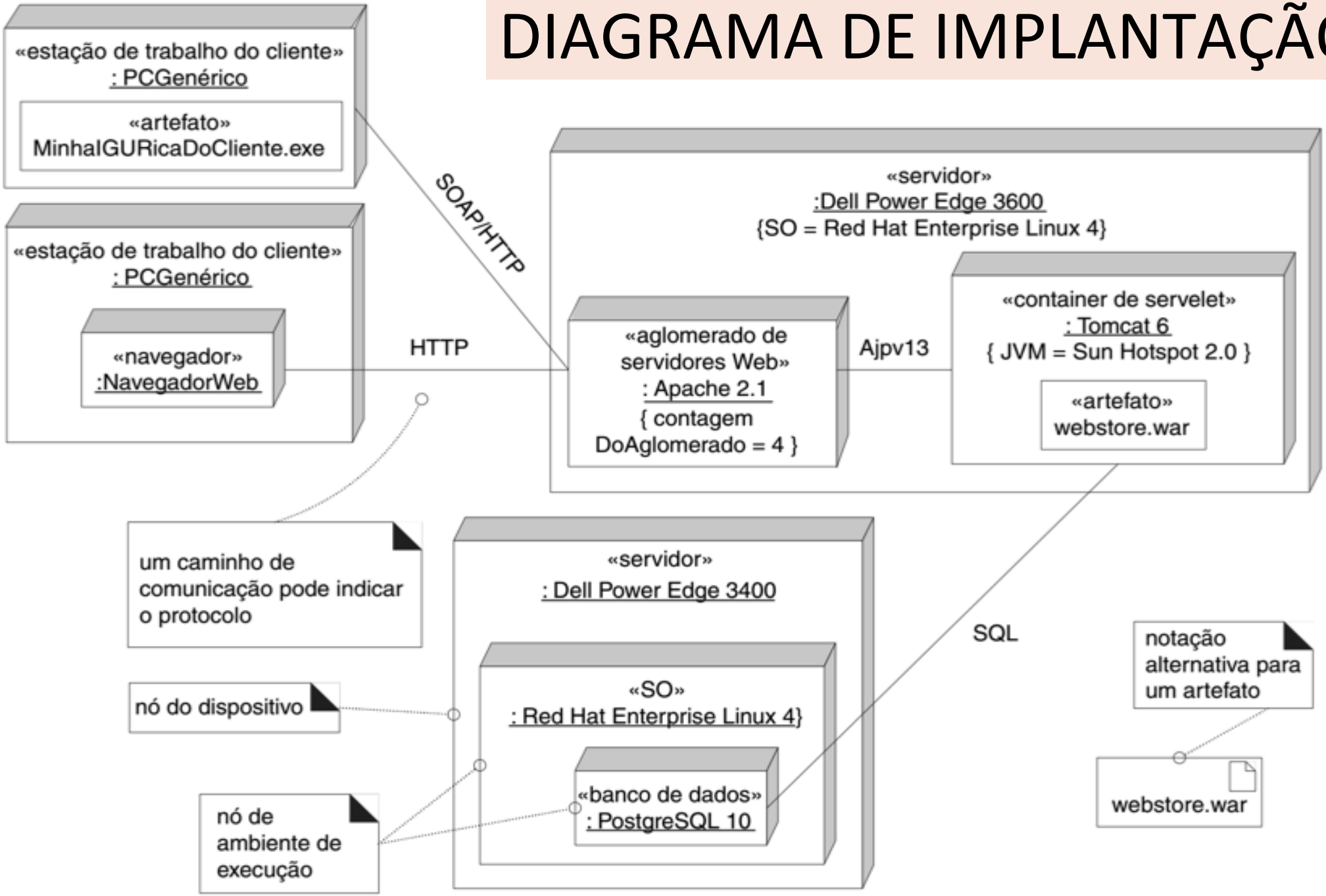
DIAGRAMA DE IMPLANTAÇÃO

Mostra a organização do hardware e a ligação do software aos dispositivos físicos.

Denota vários dispositivos de hardware e interfaces **físicas** determinadas por seu ***estereótipo***, como
(a) processador,
(b) impressora,
(c) memória,
(d) disco, e
(e) assim por diante.

Vamos a um exemplo

DIAGRAMA DE IMPLANTAÇÃO



NÓS DE DISPOSITIVOS

Nó de dispositivo (ou dispositivo) –

1. Um ***recurso computacional físico*** (por exemplo, eletrônico digital) com serviços de processamento e memória para executar software, por exemplo, um computador típico ou um telefone

NÓS DE AMBIENTE DE EXECUÇÃO

Exemplos:

1. Um **sistema operacional** (SO) é um software que hospeda e executa programas.
2. Uma **máquina virtual** (MV, como Java ou .NET VM) hospeda e executa programas.
3. Um **motor de banco de dados** (por exemplo, PostgreSQL) recebe solicitações de um programa SQL e as executa, e hospeda/executa procedimentos internos armazenados (escritos em Java ou em uma linguagem proprietária).
4. Um **navegador da Web** **hospeda e executa JavaScript**, applets Java, Flash, e outras tecnologias executáveis.
5. Um **motor de workflow**
6. Um **contêiner servlet** ou um **contêiner EJB**.

NÓS DE AMBIENTE DE EXECUÇÃO

A **conexão normal entre nós** é um caminho de comunicação, que pode ser rotulado com o protocolo. Esse geralmente indica as conexões de rede.

DIAGRAMA DE IMPLANTAÇÃO

Como a especificação UML sugere, muitos tipos de nó podem mostrar **estereótipos**, como **«servidor», «SO», «banco de dados» ou «navegador»**, mas esses não são estereótipos predefinidos oficiais da UML.

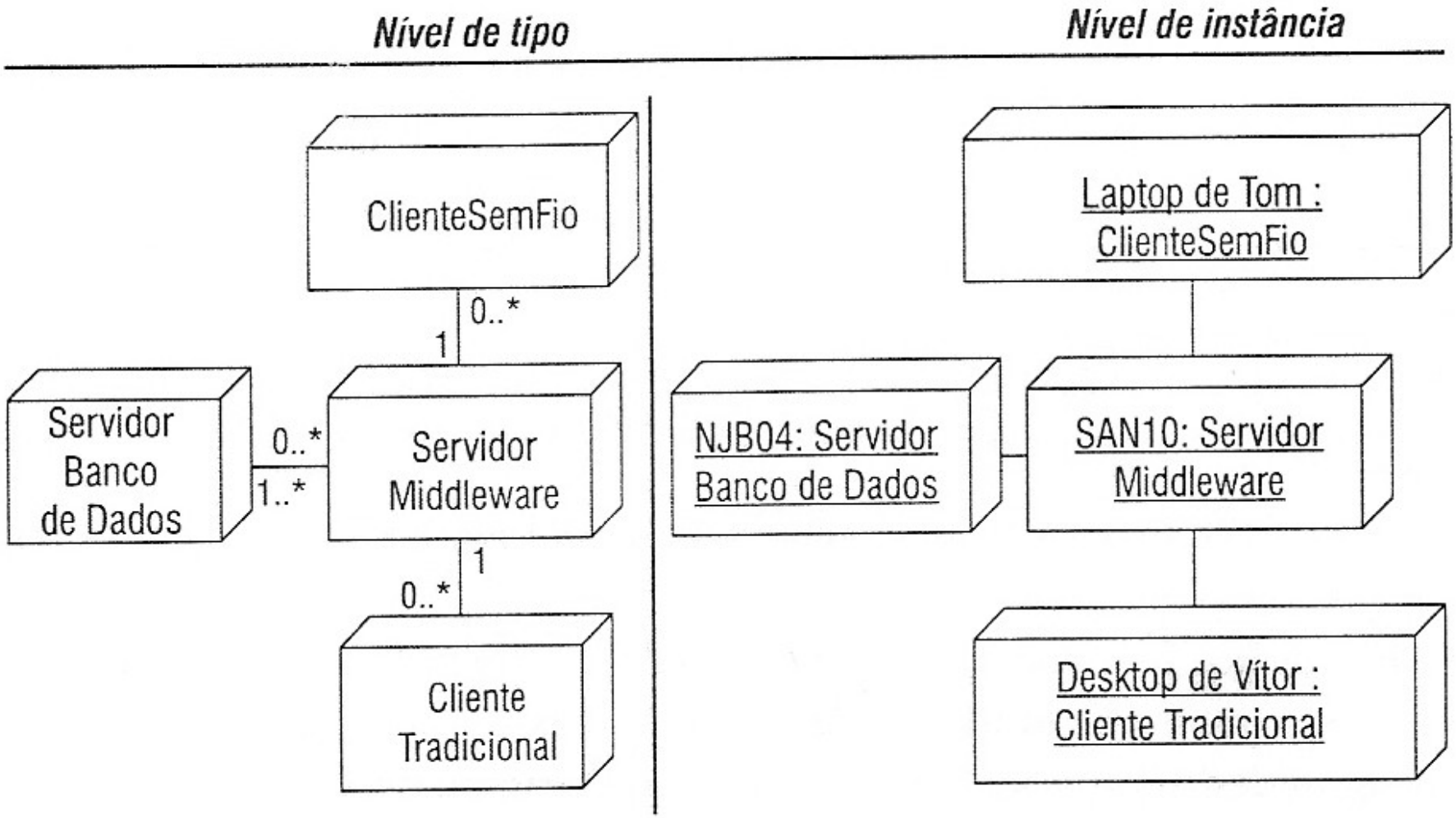
Um **nó de dispositivo** pode conter um **outro dispositivo**. **Por exemplo**, uma máquina virtual dentro de um SO dentro de um computador.

DIAGRAMA DE IMPLANTAÇÃO ***INSTÂNCIAS***

Um diagrama de implantação geralmente mostra um ***conjunto-exemplo de instâncias*** (em vez de classes).

Por exemplo, uma **instância de um computador servidor** executando uma **instância do SO Linux**

DIAGRAMA DE IMPLANTAÇÃO INSTÂNCIAS



GENERALIZAÇÃO COM NÓS

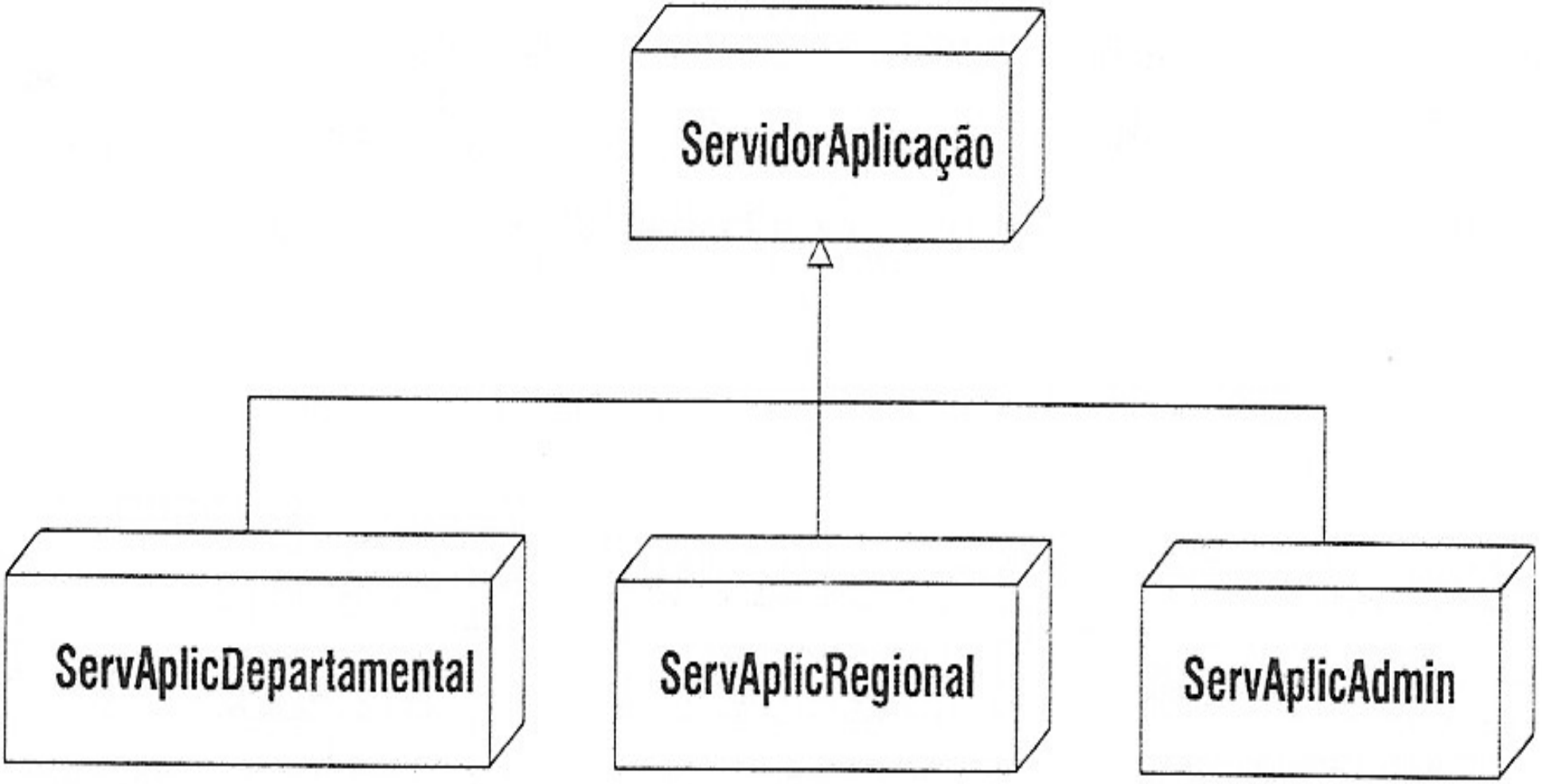
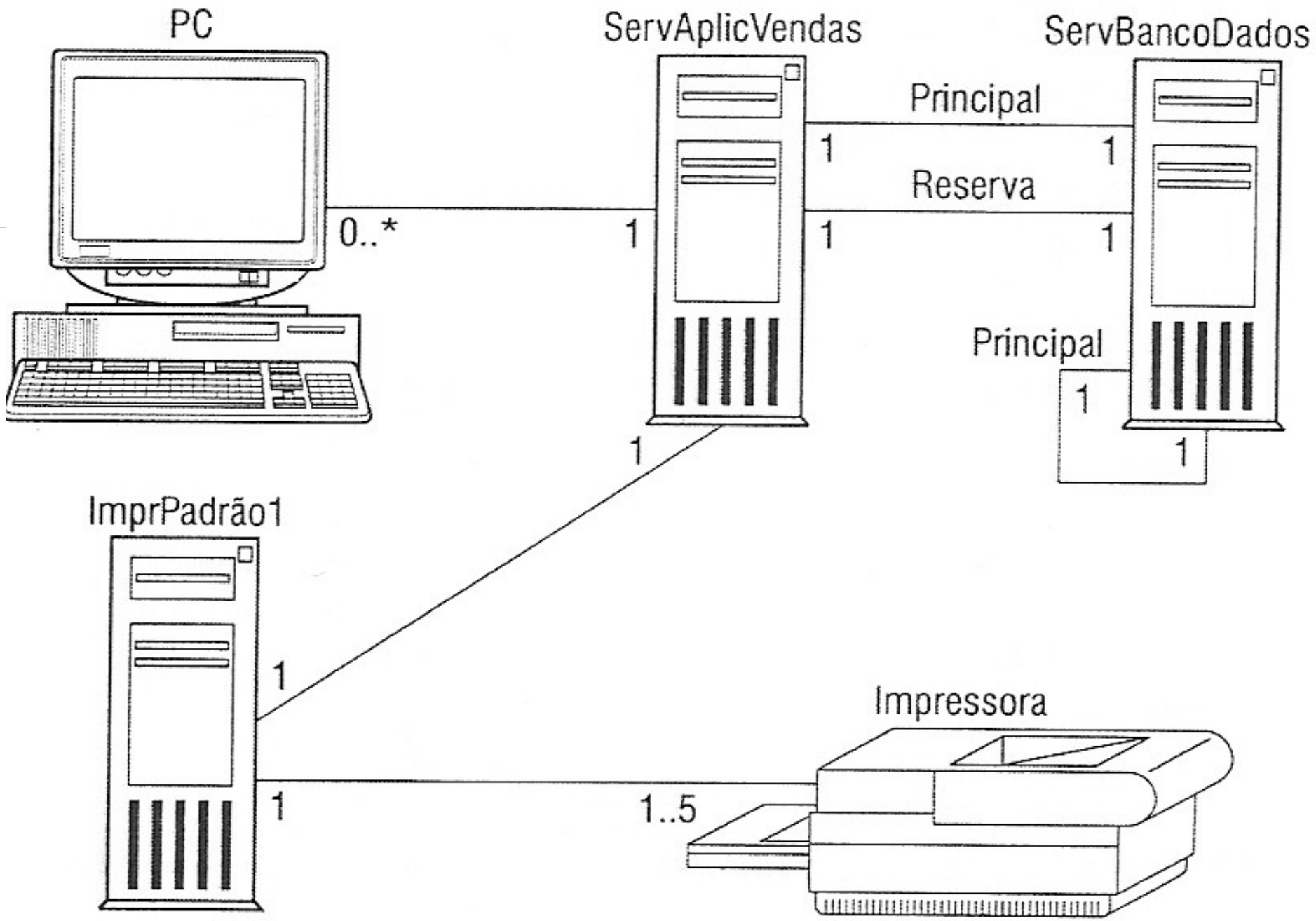
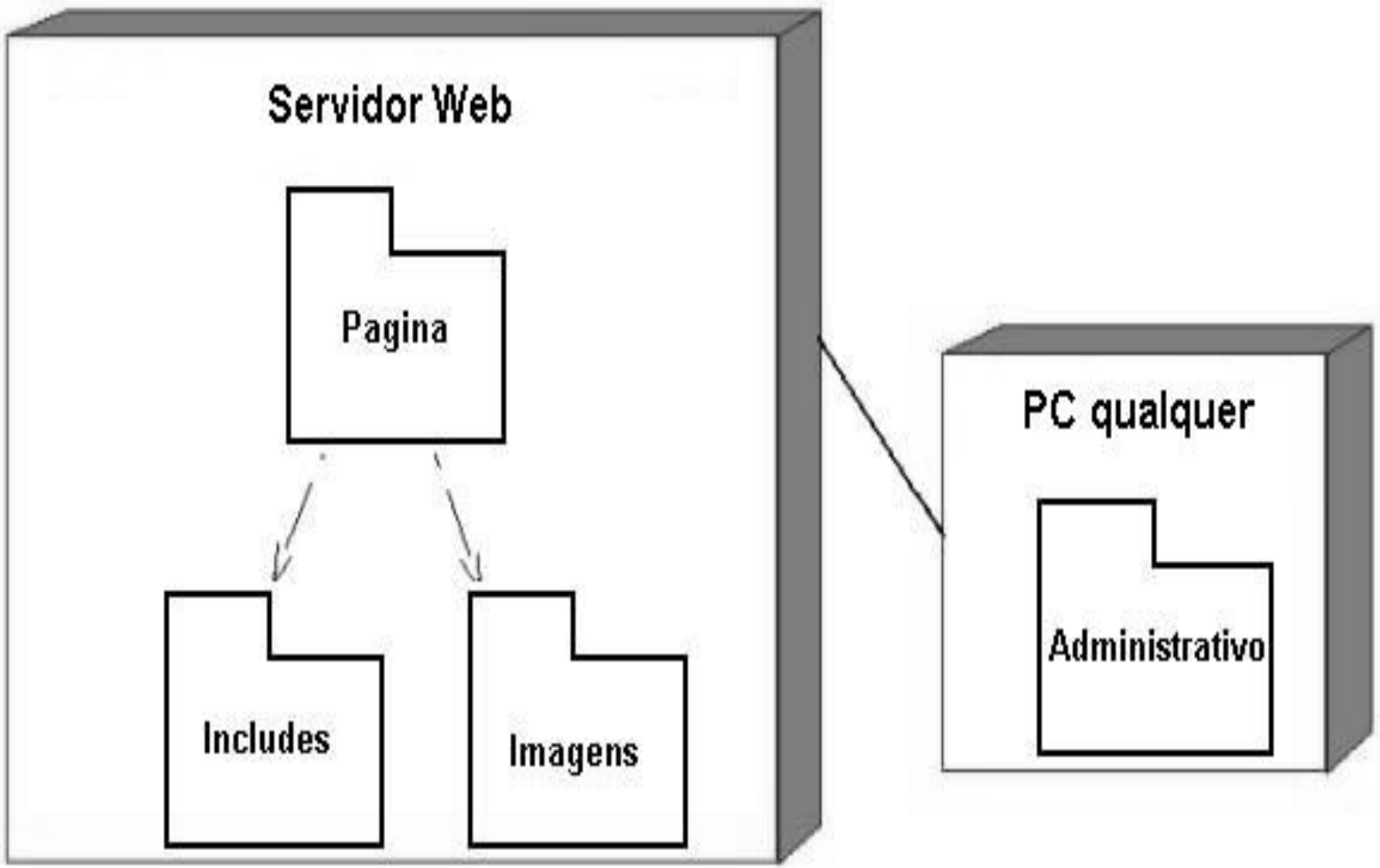


DIAGRAMA DE IMPLANTAÇÃO: TOPOLOGIA DE REDE NOTAÇÃO ALTERNATIVA



NÓS COM PACOTES



NÓS COM COMPONENTES

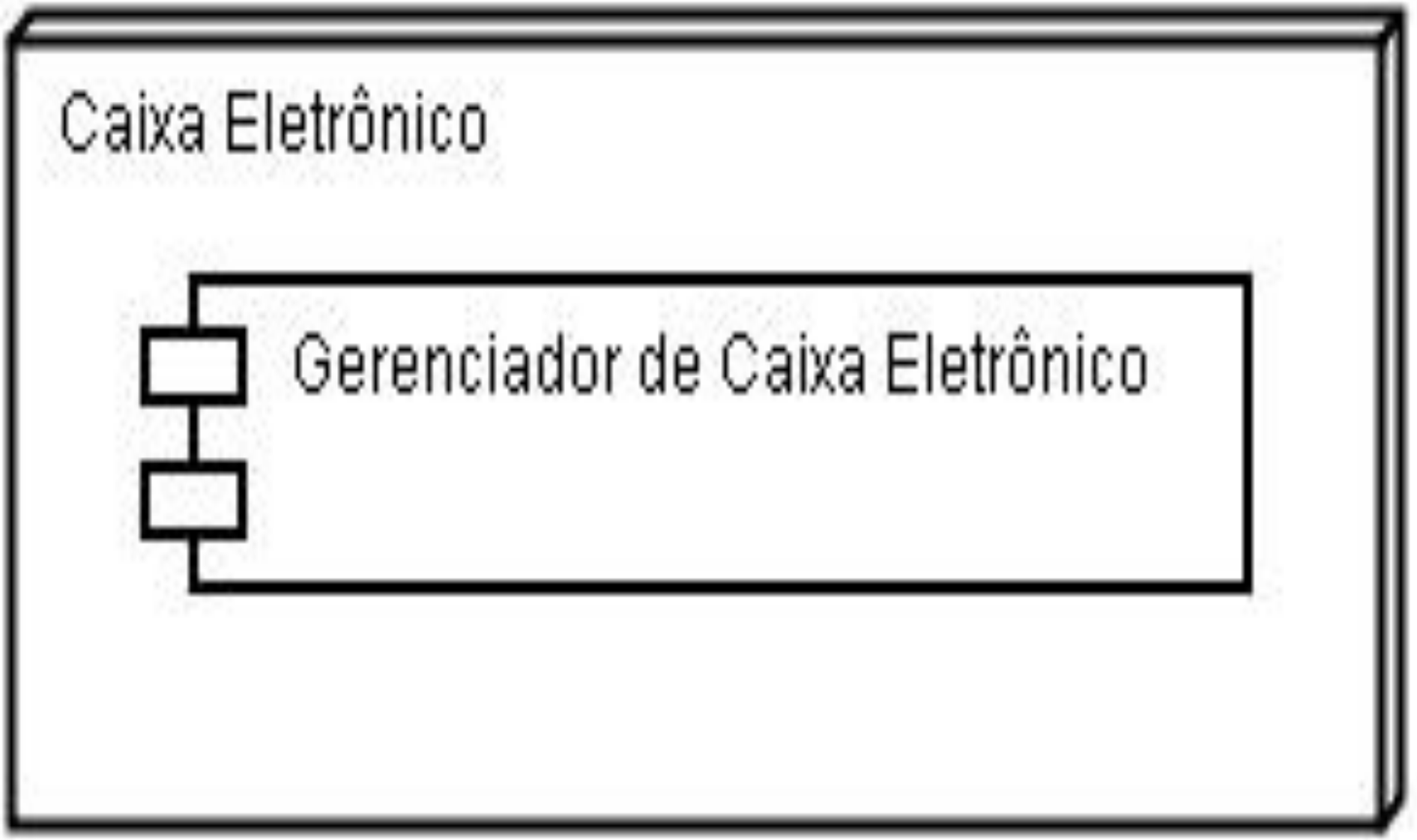


DIAGRAMA DE IMPLANTAÇÃO COM DIAGRAMA DE COMPONENTES

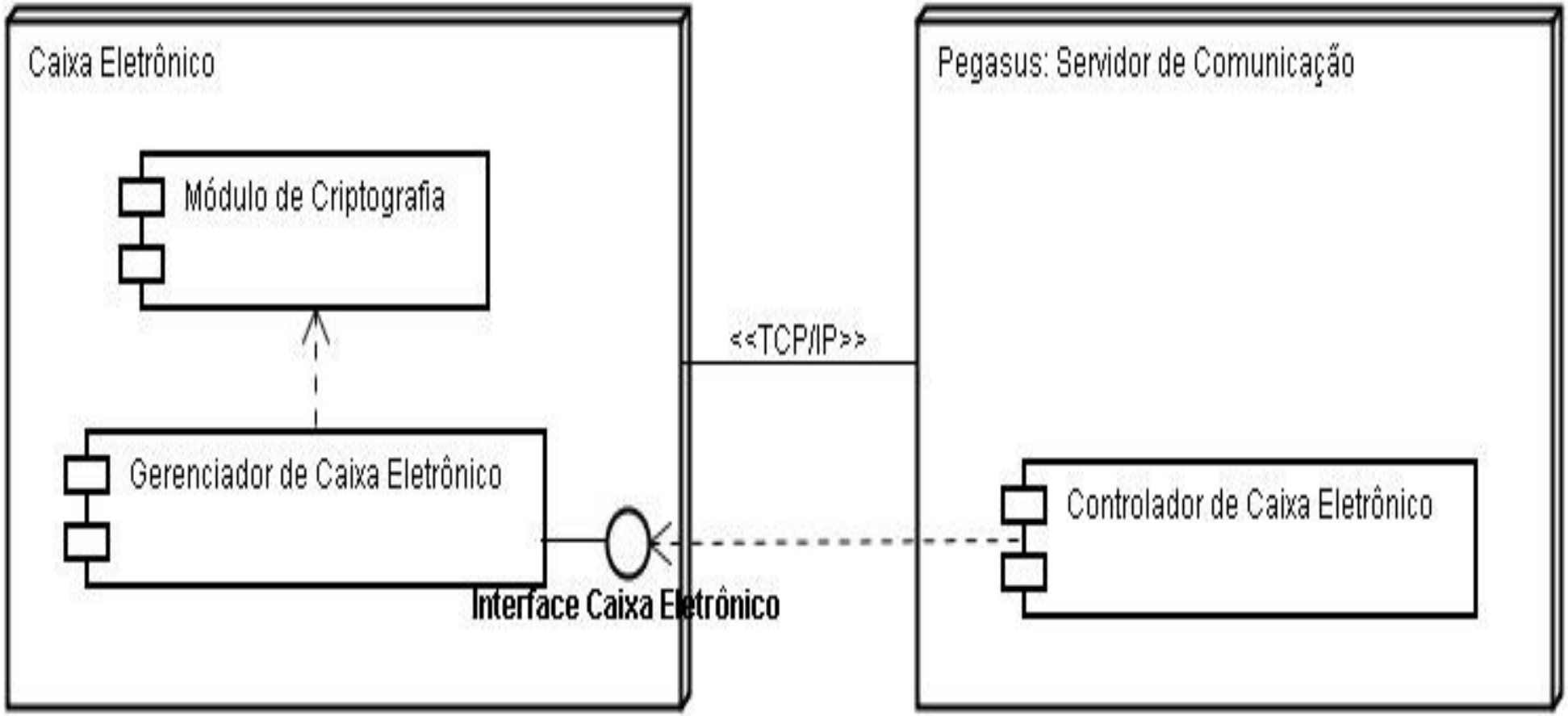


DIAGRAMA DE IMPLANTAÇÃO COM DIAGRAMA DE COMPONENTES

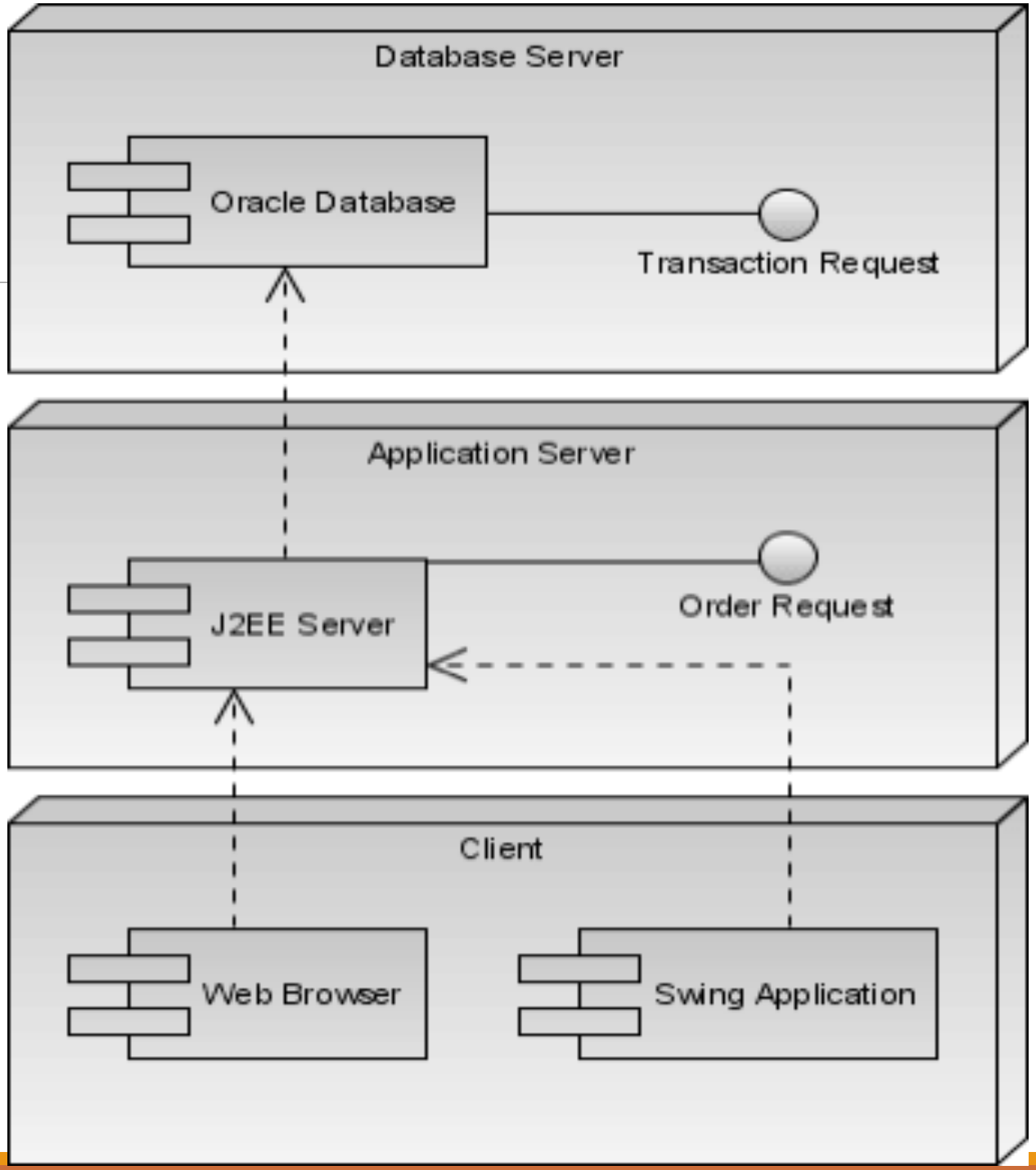
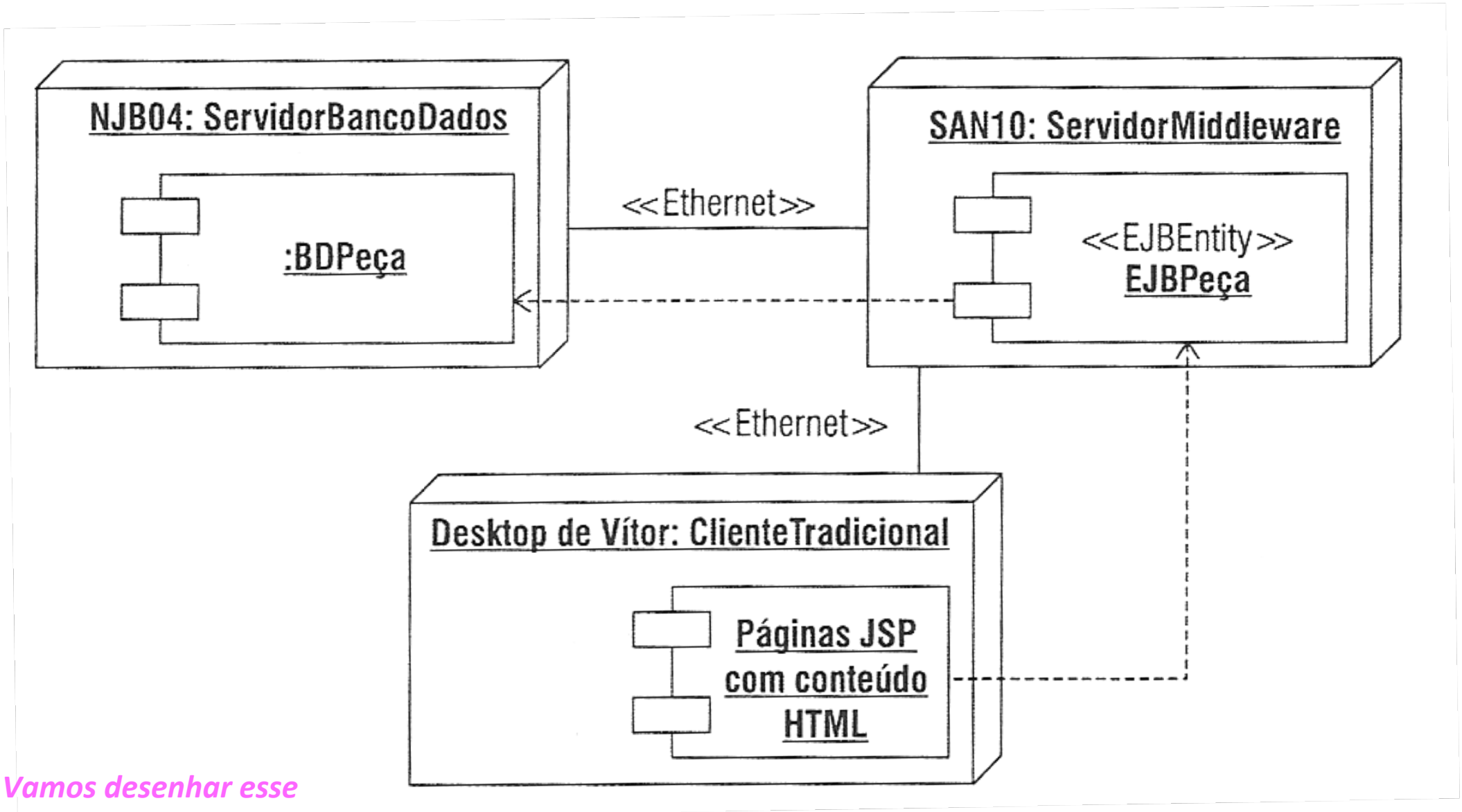


DIAGRAMA DE IMPLANTAÇÃO COM DIAGRAMA DE COMPONENTES



Vamos desenhar esse diagrama no ASTAH

DIAGRAMAS DE IMPLANTAÇÃO

Diagramas de ***implantação*** representam a arquitetura física, a qual descreve a **decomposição detalhada do hardware e software**.

Diagrama de *componente* exibe as **organizações e dependências entre componentes** .

Diagrama de implantação exibe a **organização do hardware** e a **ligação do software aos dispositivos físicos**.

Dúvidas?

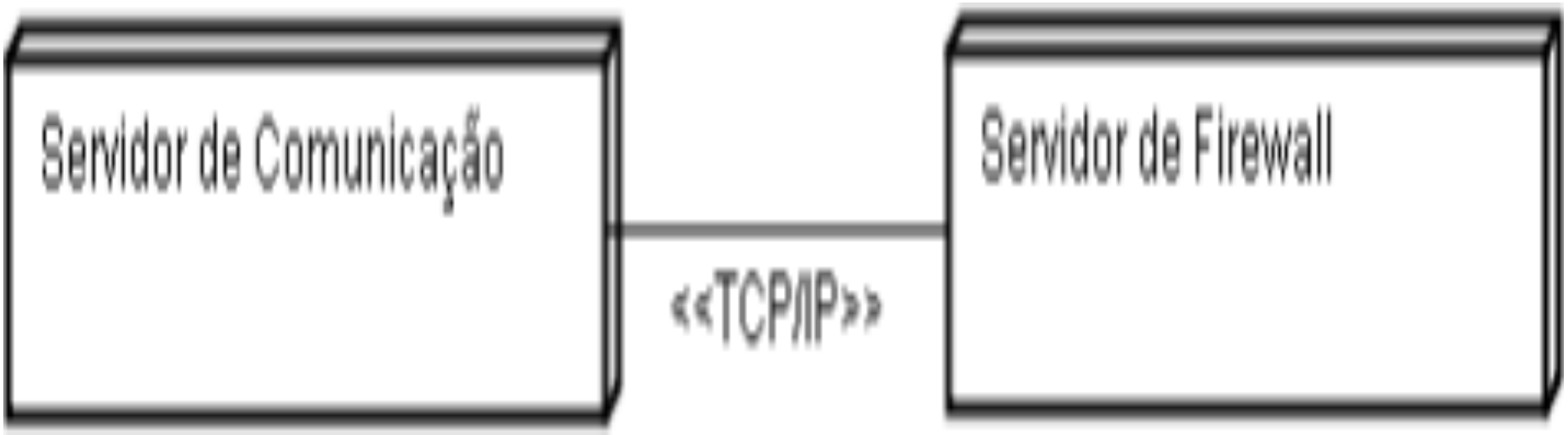
exercícios

REPRESENTE:

1. O Servidor de Comunicação comunica com o sServidor de Firewall por meio do protocolo TCP/IP.
2. O componente Módulo de Criptografia executa em um Caixa Eletrônico.
3. O Servidor de Aplicação comunica com o Servidor de *firewall* por meio do protocolo TCP/IP. O componente contas.cpp, que executa dentro do pacote Appl, no servidor de aplicação, manipula as classes conta comum, conta especial e conta poupança.

DI 1 - RESPOSTA

O servidor de comunicação comunica com o servidor de *firewall* por meio do protocolo TCP/IP.



DI 2 - RESPOSTA

O componente Módulo de Criptografia roda em um caixa eletrônico.

DI 2 - RESPOSTA

1. O servidor de aplicação se comunica com o servidor de firewall através do protocolo TCP/IP. O componente contas.cpp, que roda dentro do pacote Appl, no servidor de aplicação, manipula as classes conta comum, conta especial e conta poupança.

DI 2 - RESPOSTA