Projeto de Sistemas de Informação

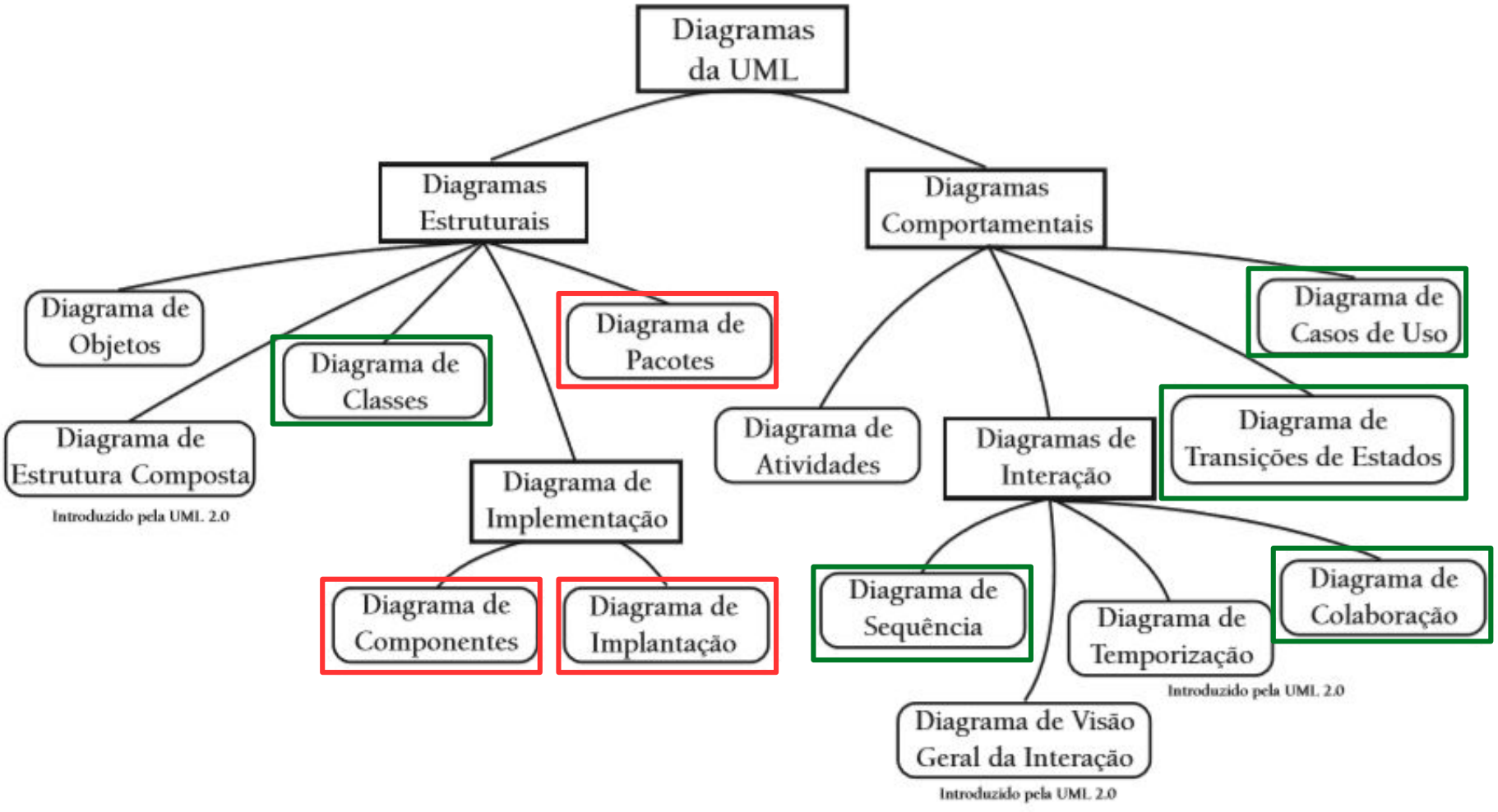
Pacotes, Componentes e Implantação

Lesandro Ponciano

2024

Objetivos da Aula

- Contextualizar os diagramas UML no projeto de software
  - Diagrama de pacote
  - Diagrama de componentes
  - Diagrama de implantação
- Discutir
  - Conceitos
  - Notações
  - Exemplos de aplicação

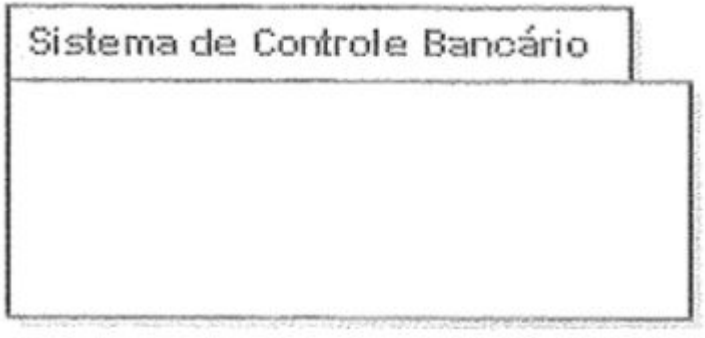

Diagramas da UML
Diagramas Estruturais
Diagramas Comportamentais
Diagrama de Objetos
Diagrama de Classes
Diagrama de Pacotes
Diagrama de Estrutura Composta
Introduzido pela UML 2.0
Diagrama de Implementação
Diagrama de Componentes
Diagrama de Implantação
Diagrama de Casos de Uso
Diagrama de Atividades
Diagramas de Interação
Diagrama de Transições de Estados
Diagrama de Sequência
Diagrama de Temporização
Introduzido pela UML 2.0
Diagrama de Colaboração
Diagrama de Visão Geral da Interação
Introduzido pela UML 2.0

1 Diagrama de Pacotes

- Descreve como os elementos do modelo estão organizados em pacotes e demonstra as dependências entre eles
- Útil na modelagem de subsistemas e modelagem de subdivisões da arquitetura
- Também pode representar
  - conjunto de sistemas integrados, representados por pacotes
  - submódulos englobados pelo sistema

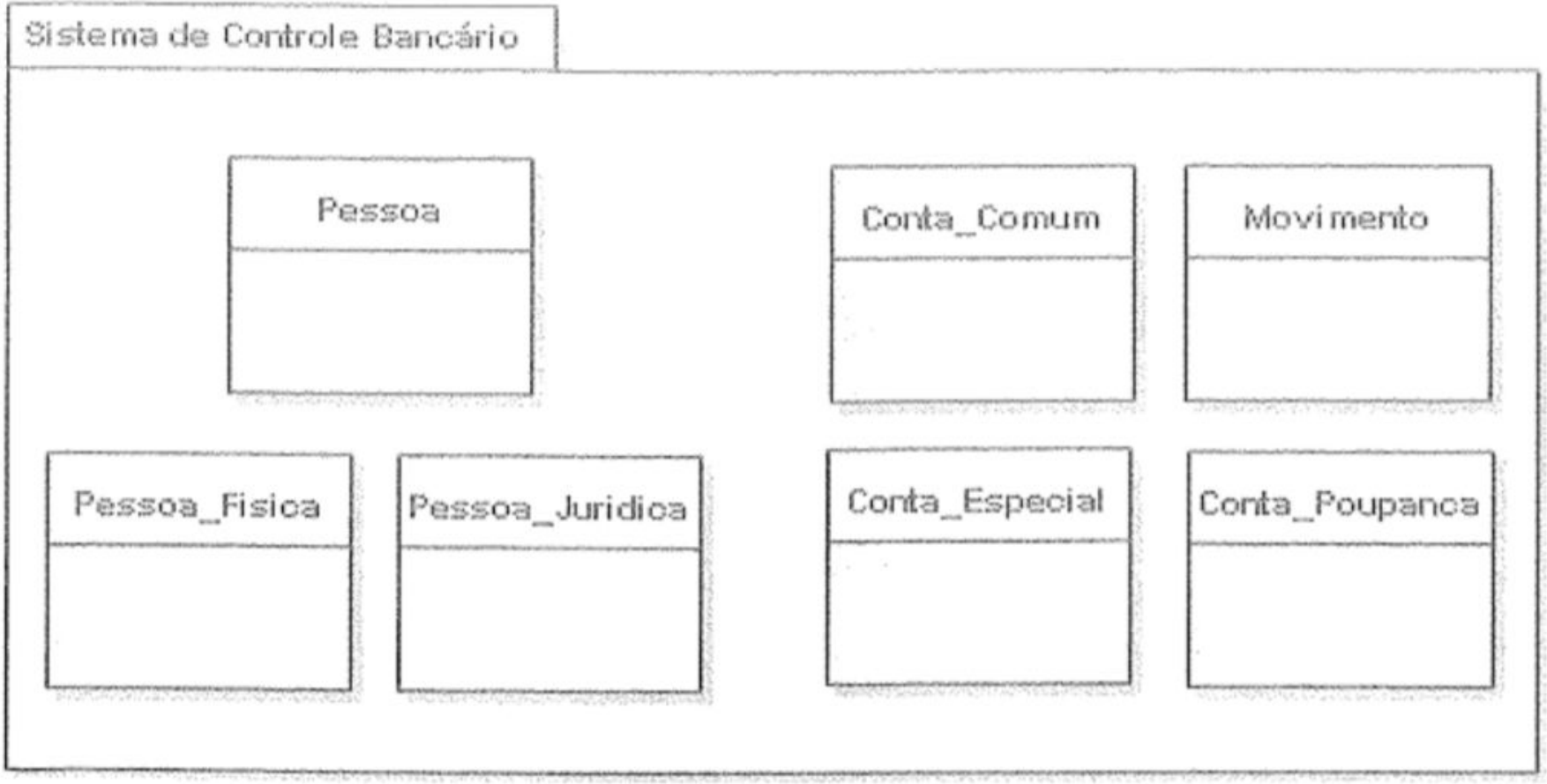
Pacotes

- São usados para agrupar elementos e fornecer denominações para esses grupos
- Pode representar
  - Sistema
  - Subsistema
  - Biblioteca
  - Etapas de um processo
  - Outros agrupamentos



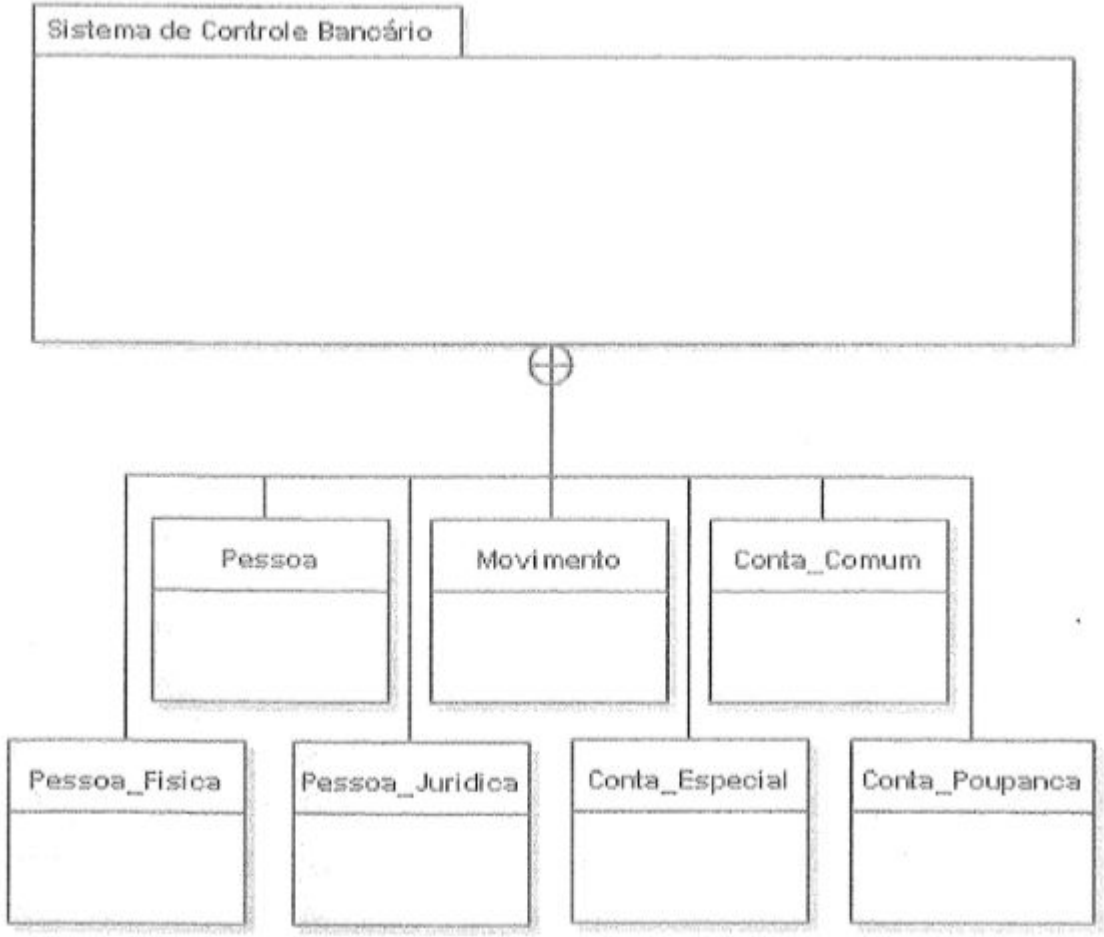
Exemplo de representação de um pacote sem revelar o seu conteúdo

Pacote e seu Conteúdo

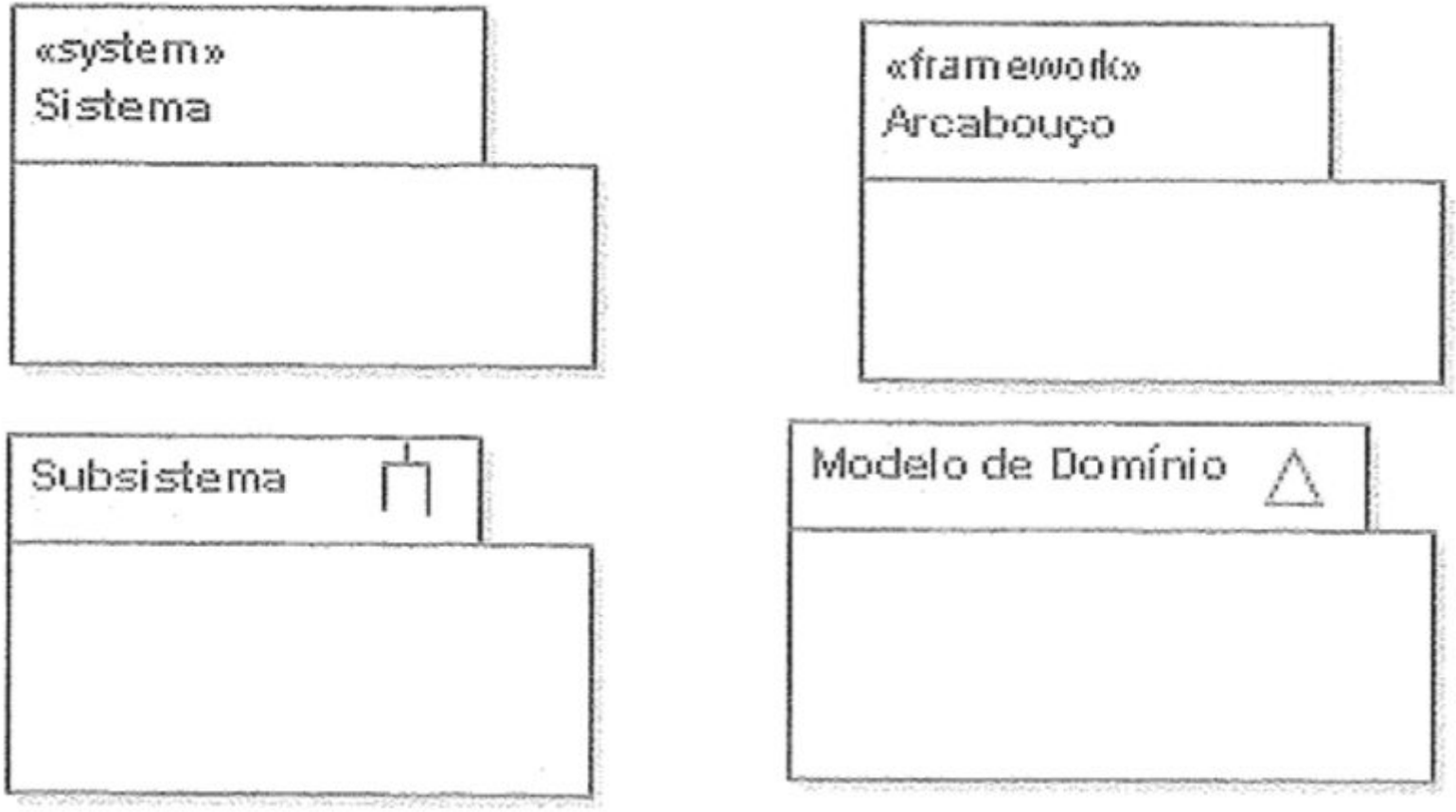


O nível de detalhes do conteúdo depende do que se deseja representar/indicar

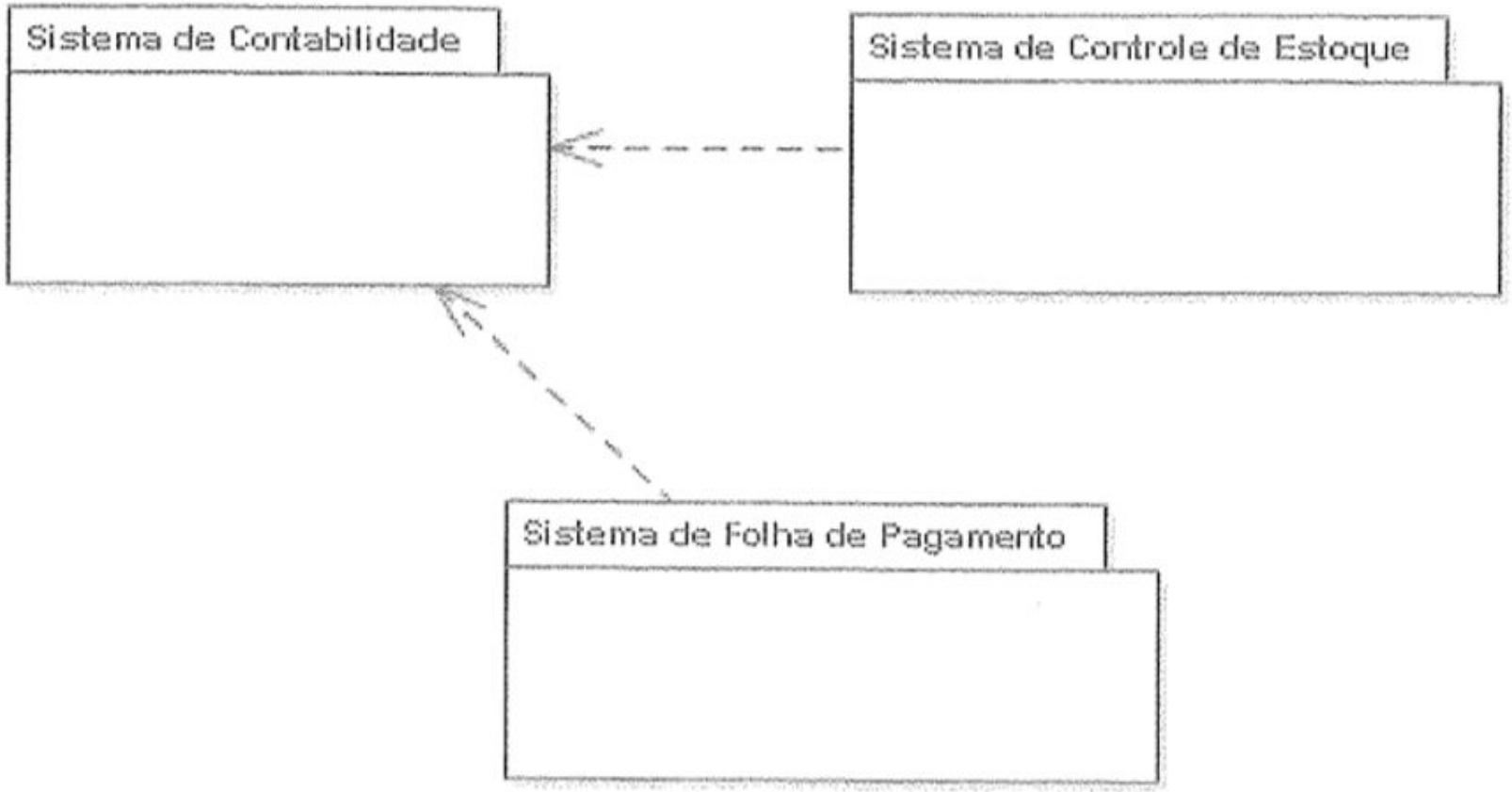
Representação Alternativa



Estereótipos Aplicados a Pacotes



Dependências entre Pacotes



O pacote dependente necessita de alguma forma do pacote do qual depende

Dependências entre Pacotes

- As dependências entre pacotes pode ser de dois tipos *merge* e *import*

- Estereótipo <<*merge*>>
  - os elementos do pacote que utiliza essa dependência serão unidos aos elementos do outro pacote

- Estereótipo <<*import*>>
  - o pacote que utiliza essa dependência está importando alguma característica ou elemento do outro pacote

2 Diagrama de Implementação

- O diagrama de implementação da UML é utilizado para representar a arquitetura física de um sistema

- O modelo construído a partir desse diagrama é denominado
  - modelo de implementação
  - modelo da arquitetura física

2.1 Componentes

2.2 Implantação

2.1 Diagrama de Componentes

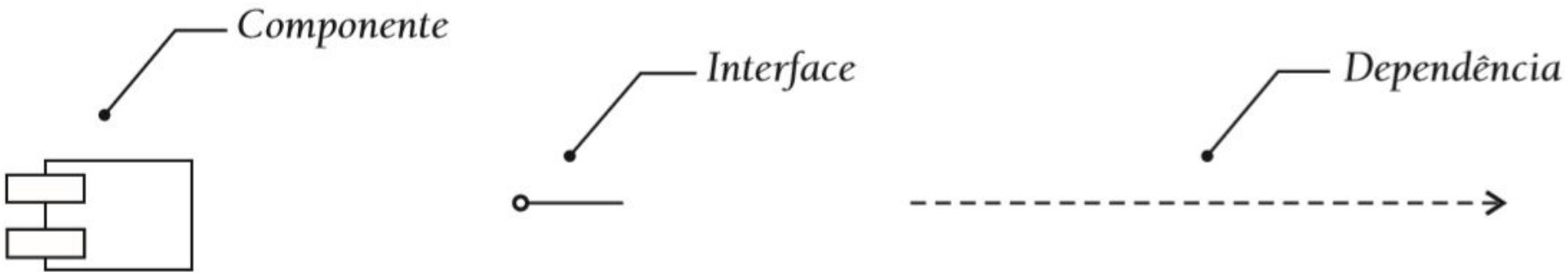
- O diagrama de componentes identifica os componentes que fazem parte de
  - um sistema
  - um subsistema
  - ou mesmo classes internas de um componentes individual

- Um componente pode representar
  - Componente lógico: De negócio ou processo
  - Componente físico: Como arquivos contendo código-fonte, arquivos de ajuda, bibliotecas, arquivos de executáveis etc

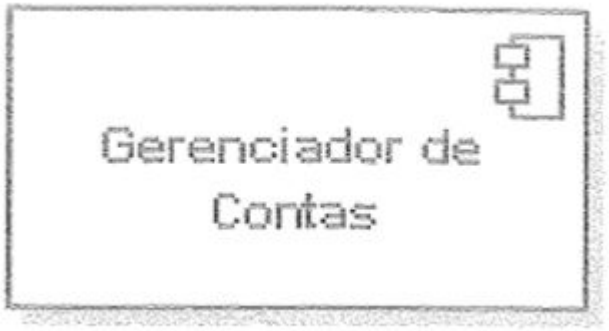
Componentes

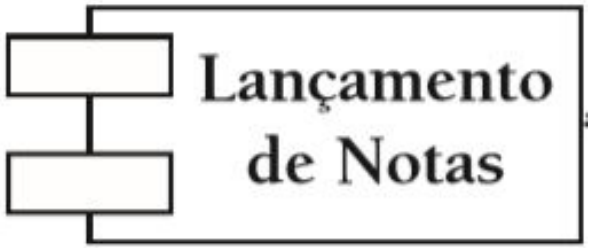
- Um componente é uma unidade autônoma dentro de um sistema ou subsistema
- Podem conter:
  - Interfaces fornecidas
  - Interfaces requeridas
- Seus interiores são transparentes e inacessíveis por outro meio que não seja fornecido por suas interfaces

Elementos do Diagrama



Exemplos de Componentes

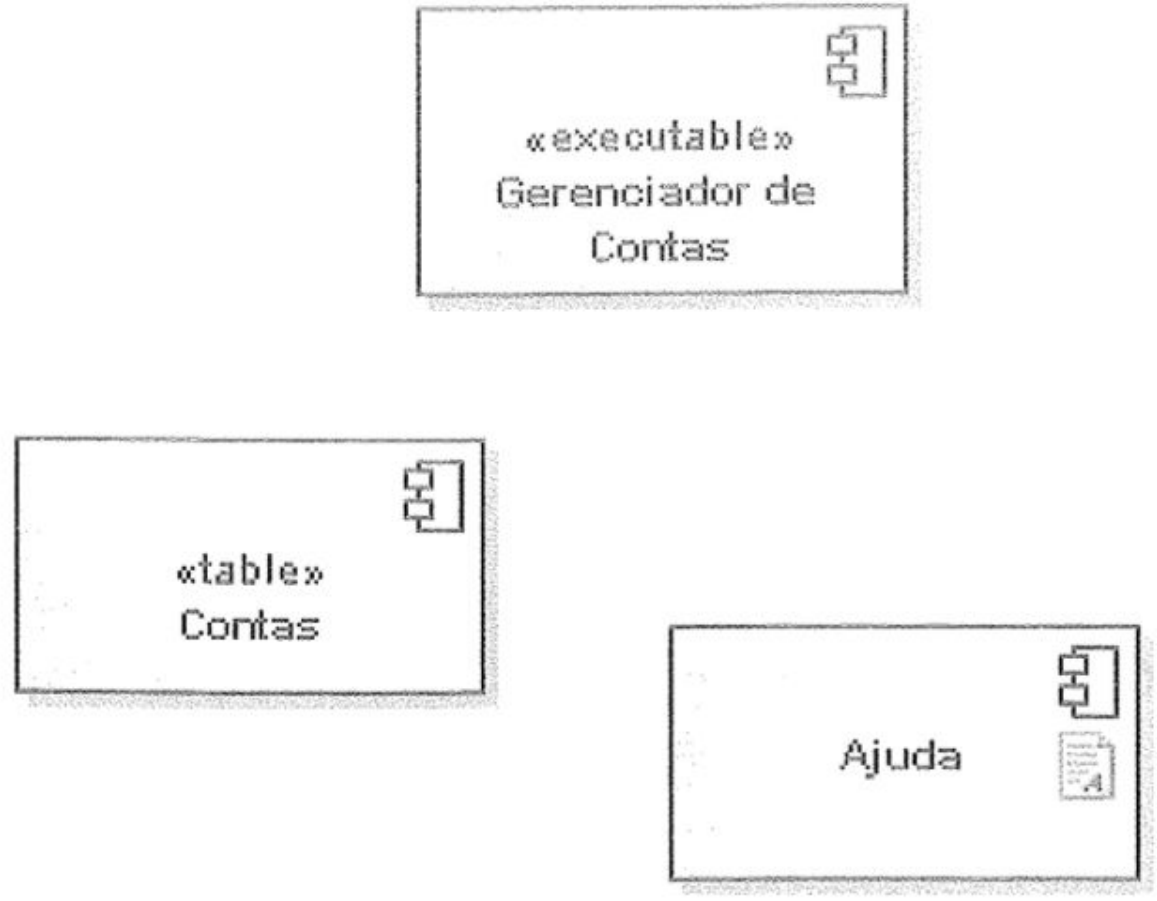




Estereótipos

- Há diversos Estereótipos definidos na UML para componentes
  - Executável (*executable*)
  - Biblioteca (*library*)
  - Tabela (*table*)
  - Documento (*document*)
  - Arquivo (*file*)

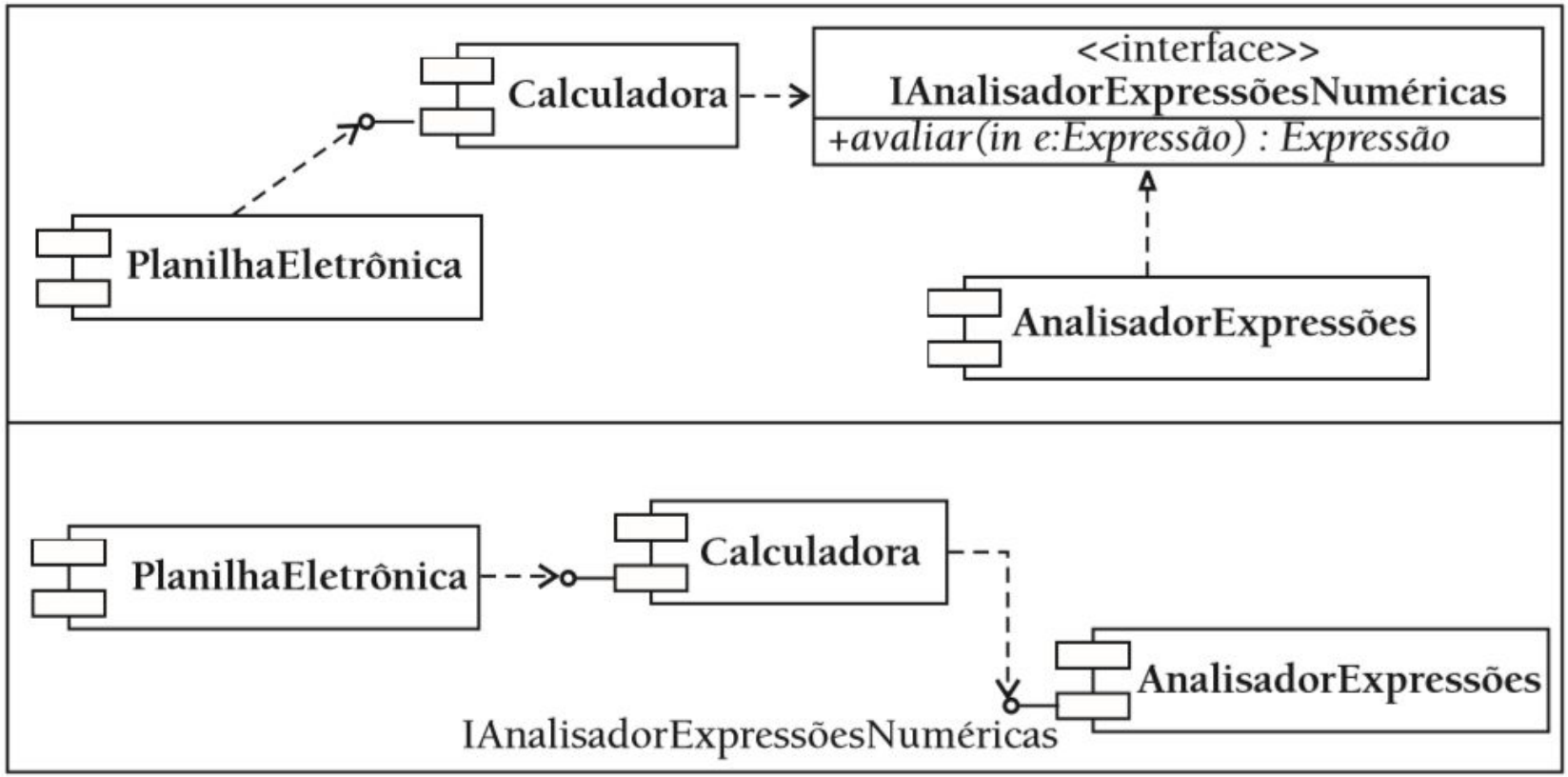
Componentes e Estereótipos



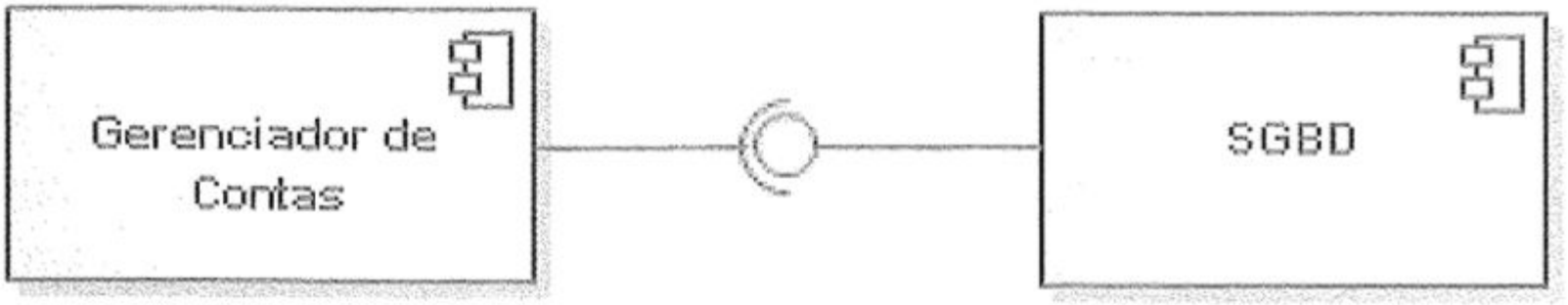
Interfaces

- Uma interface representa um serviço realizado por uma classe ou componente
- As interfaces não possuem implementação ou qualquer especificação interna
- Se um componente implementa uma interface, este relaciona-se com ela através de uma realização
- Se um componente utiliza a interface, há um relacionamento de dependência

Alternativas de Representação de Interfaces

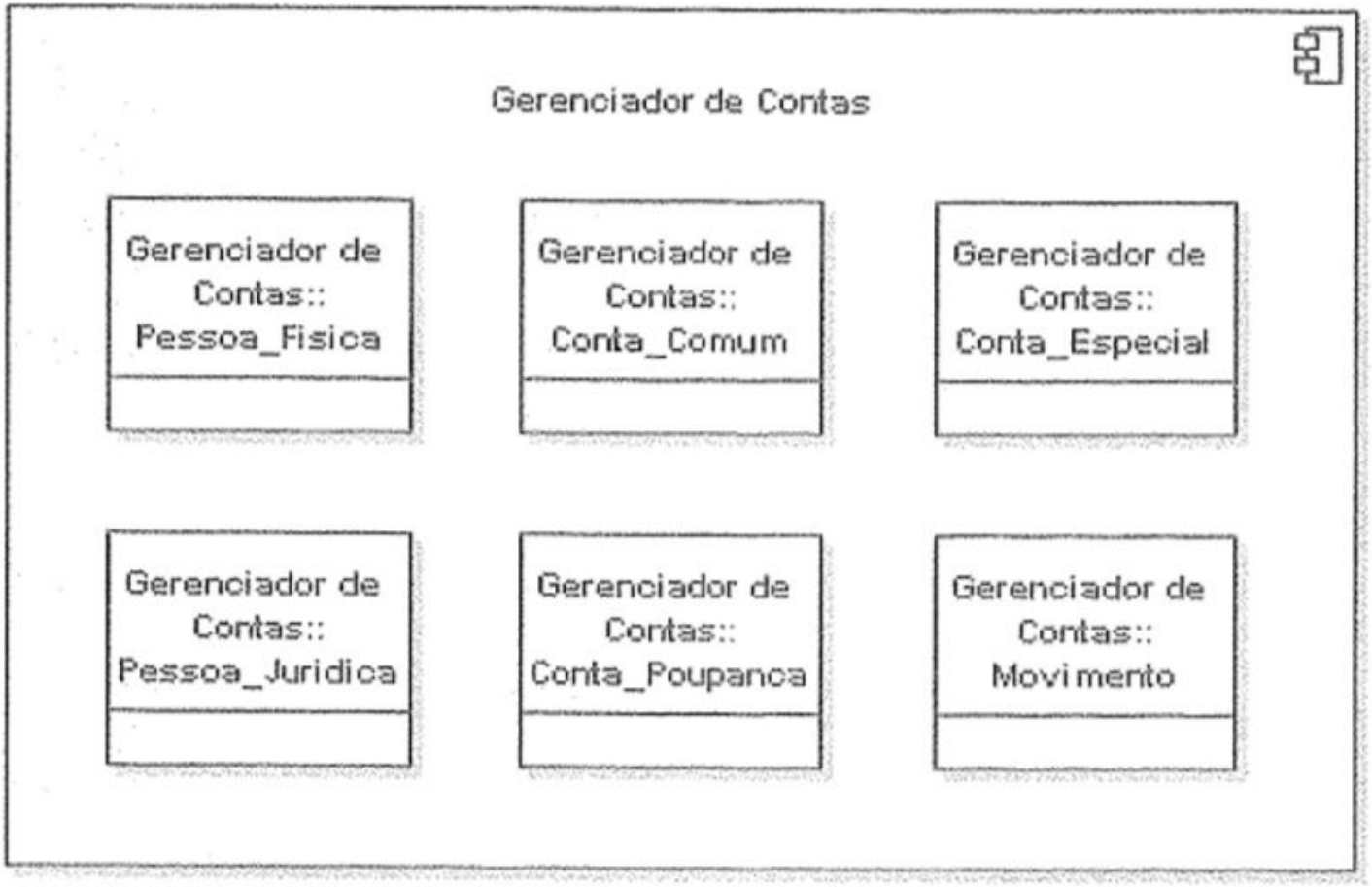


Interfaces Requeridas e Fornecidas

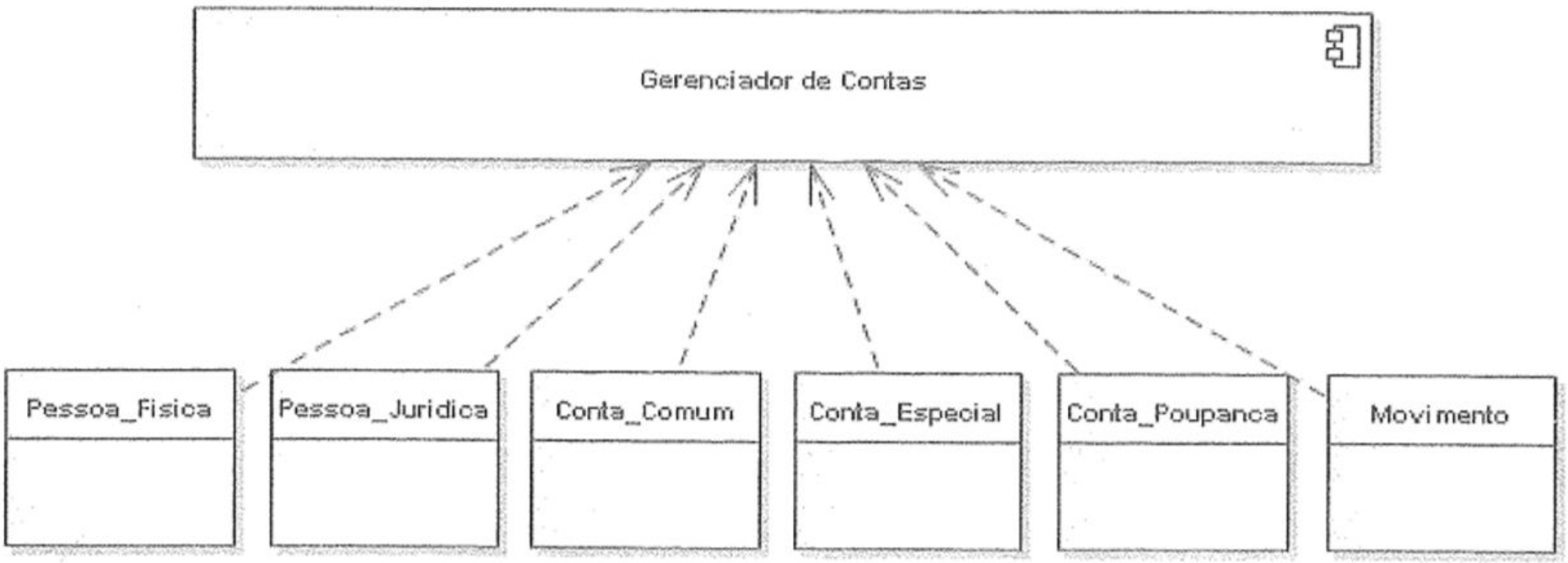


- O componente "Gerenciador de Contas" tem uma interface requerida com o componente "SGBD"
- O componentes "SGBD" tem uma interfaces fornecida com o componente "Gerenciador de Contas"

Classes e Componentes Internos

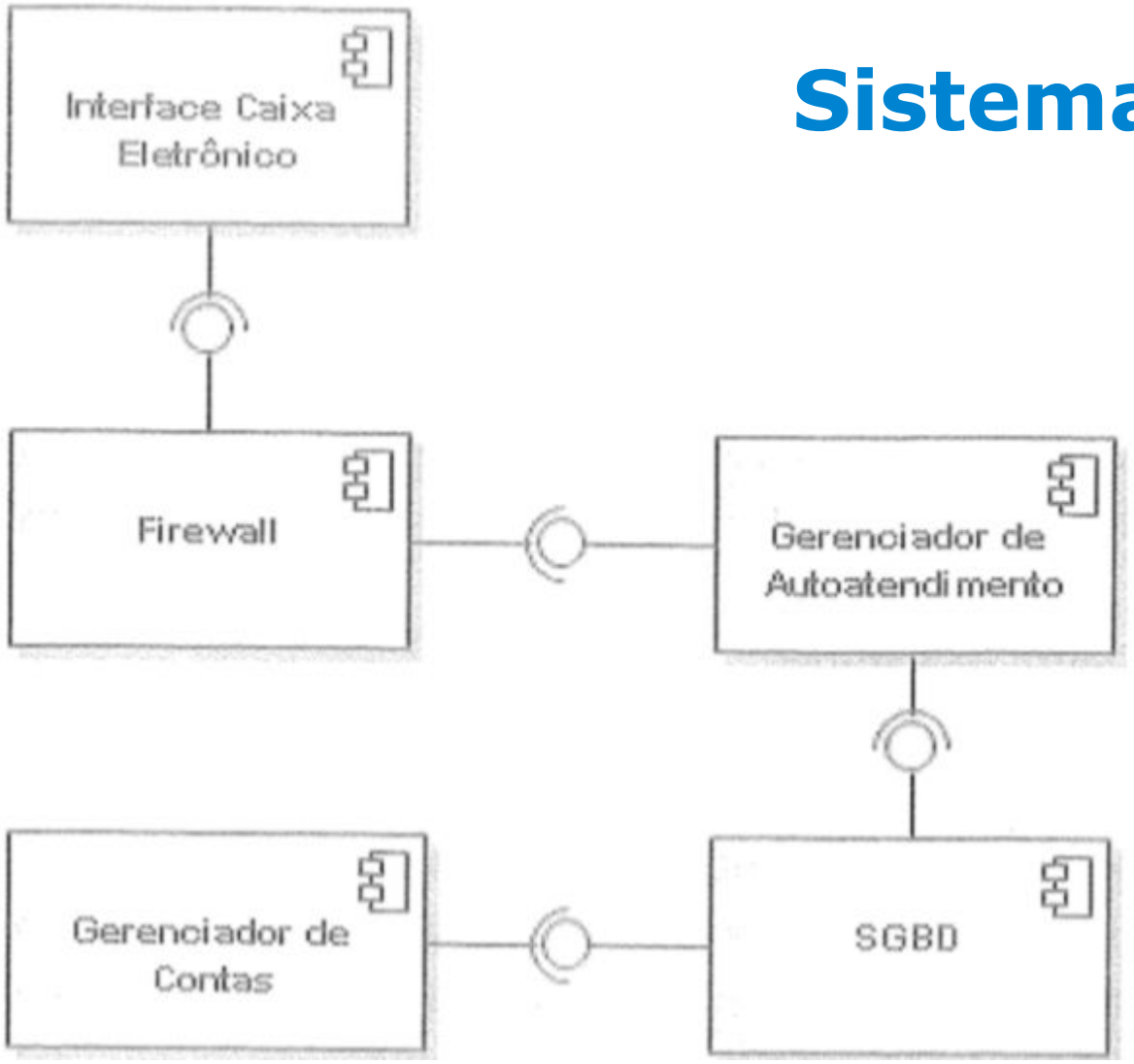


Outra forma de Representação

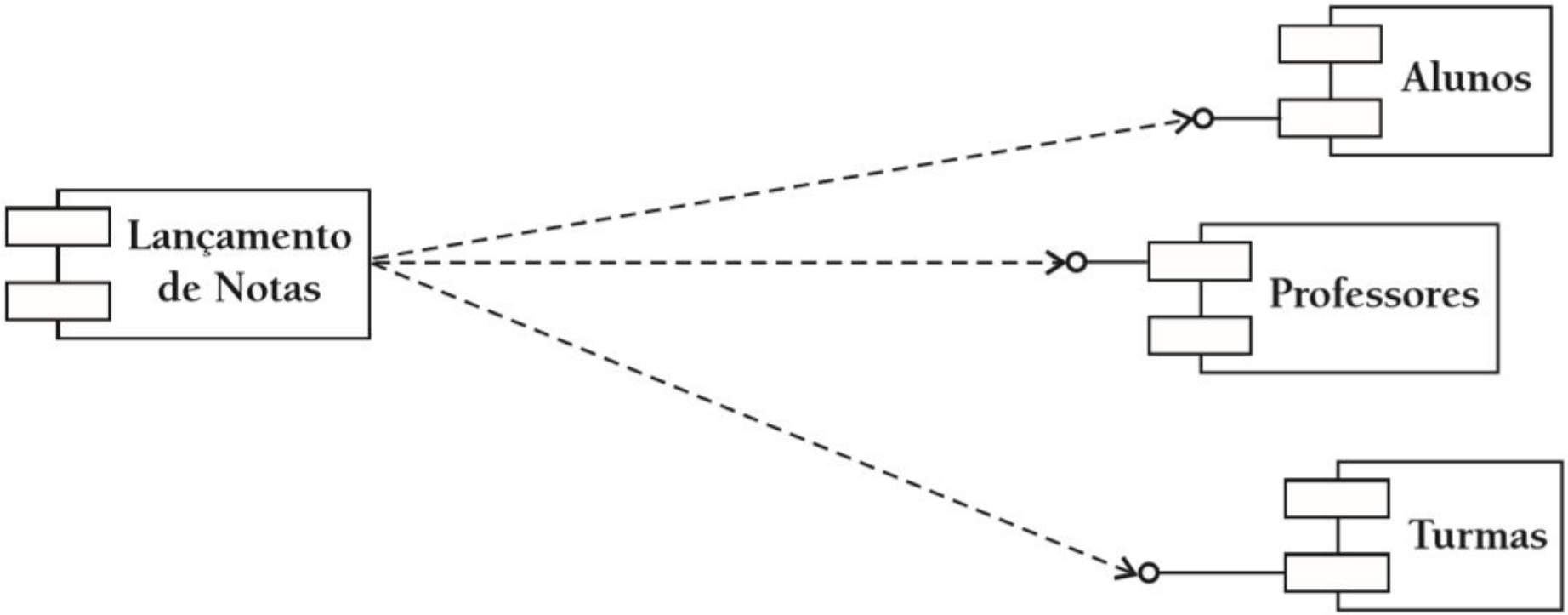


Outra forma de representar as classes internas de um componente é por meio do relacionamento de dependência

Exemplo
Sistema de Controle Bancário



Exemplo de Sistema Acadêmico

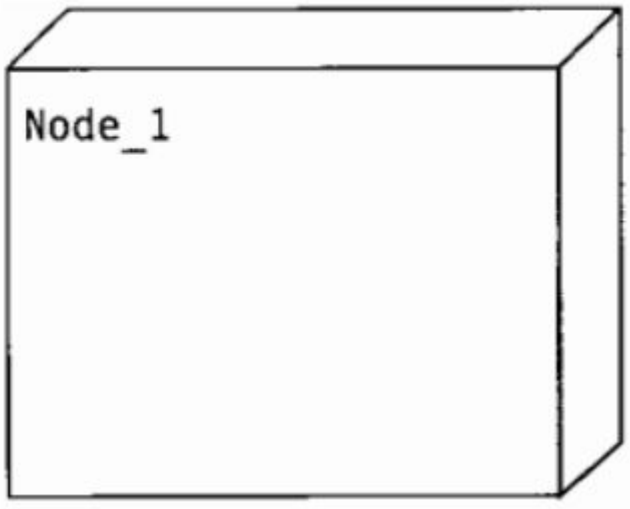


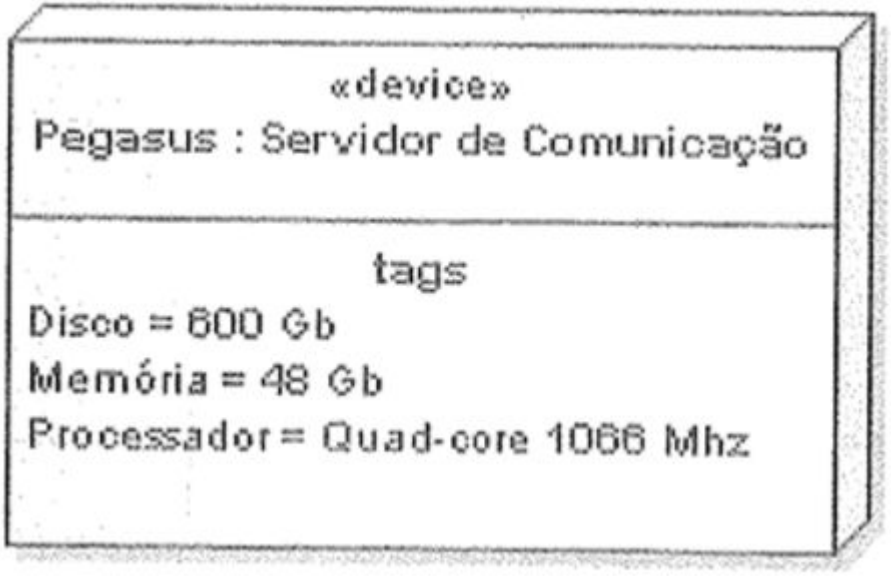
2.2 Diagrama de Implantação

- Enfoca a questão da organização da arquitetura física sobre a qual o software será implantado e executado em termos de hardware
  - Computadores, pessoas, servidores, *smartphones*, etc
- Enfoca a forma como as máquinas estarão conectadas e por meio de qual protocolo elas se comunicarão e transmitirão informações
- Permite identificar como se dá a distribuição dos módulos de um sistema em situações em que há mais de um recurso computacional (chamado nó)
  - Se há apenas um recurso, o diagrama não é necessário

Nó

- Cada nó é uma máquina física



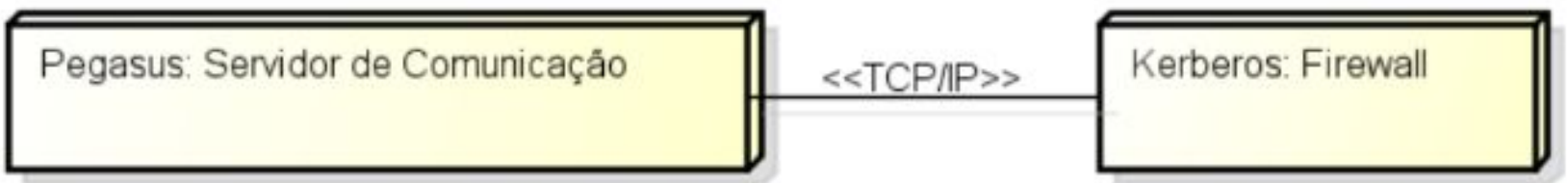


Estereótipos

- <<device>>, <<computer>>,<<server>>,<<storage>>

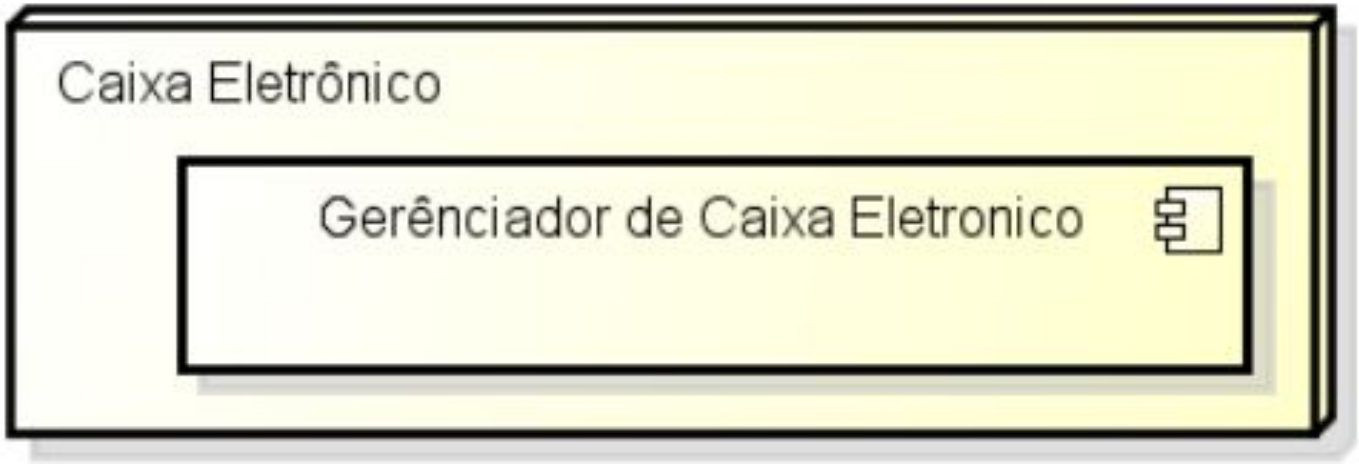
Associações entre Nós

- Nós podem ter associações entre eles
- Se dois nós possuem associação entre eles, então há uma forma de comunicação e troca de informações
- Associações podem ter estereótipos



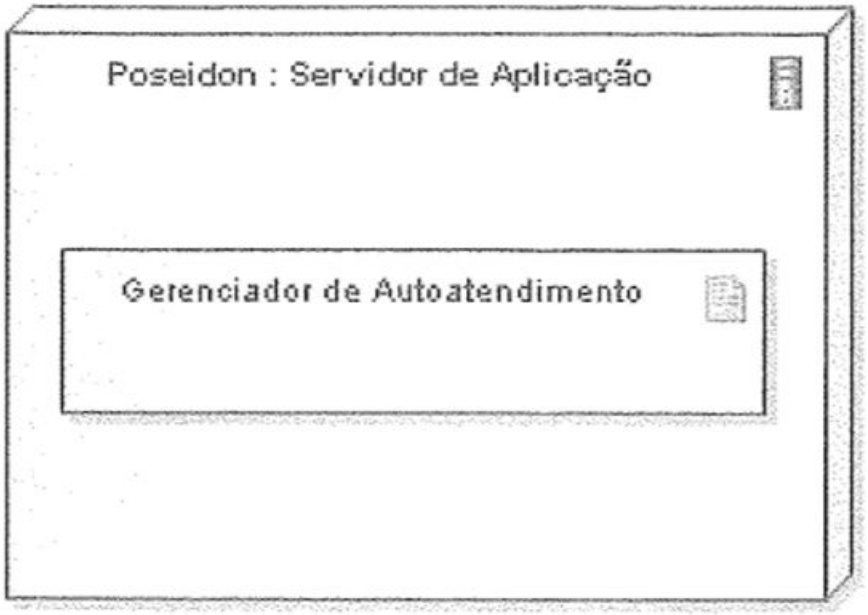
Nós contendo Componentes

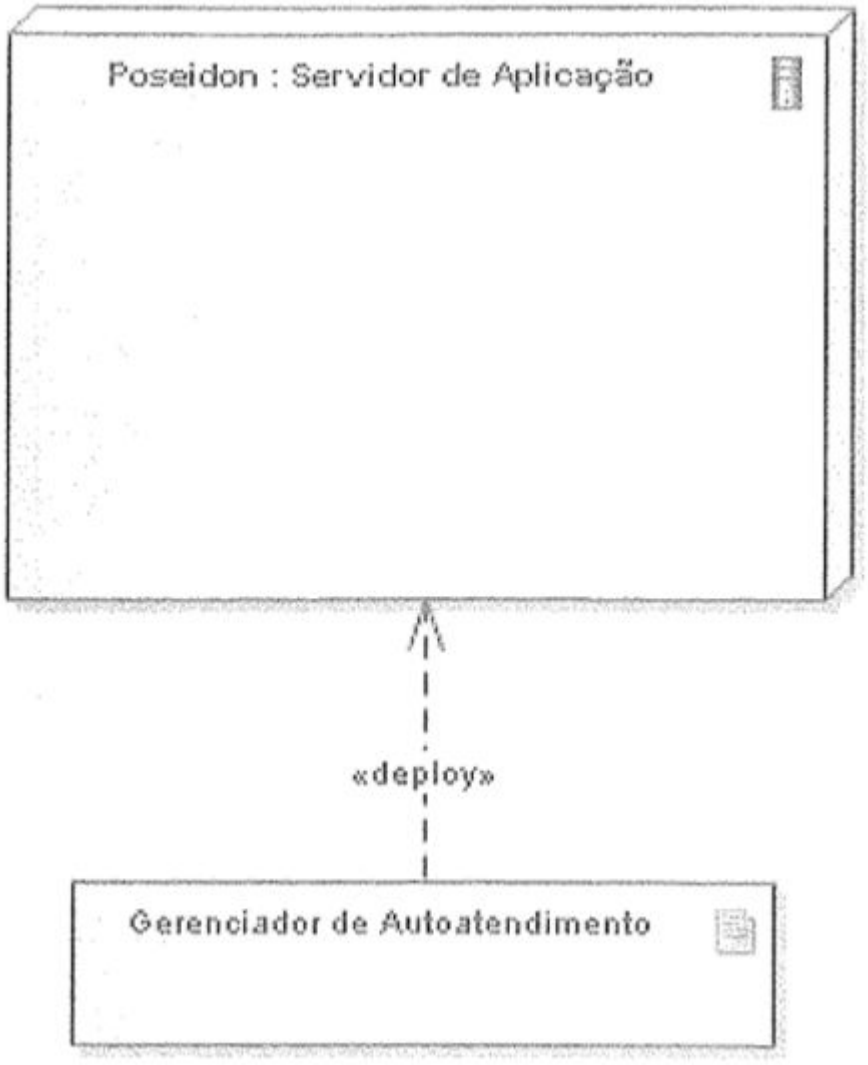
- Identificação de quais componentes executam em determinado nó



Artefatos

- Um artefato, assim como o nó, é
  - uma entidade física
  - elemento concreto que existe no mundo real
- Um artefato pode ser código fonte, um arquivo executável, um documento de texto
- Um artefato pode ser uma manifestação, no mundo real, de um componente
  - Um componente "Gerenciador de autoatendimento" pode ser um arquivo de código fonte
  - Nem sempre existirá um artefato para cada componente

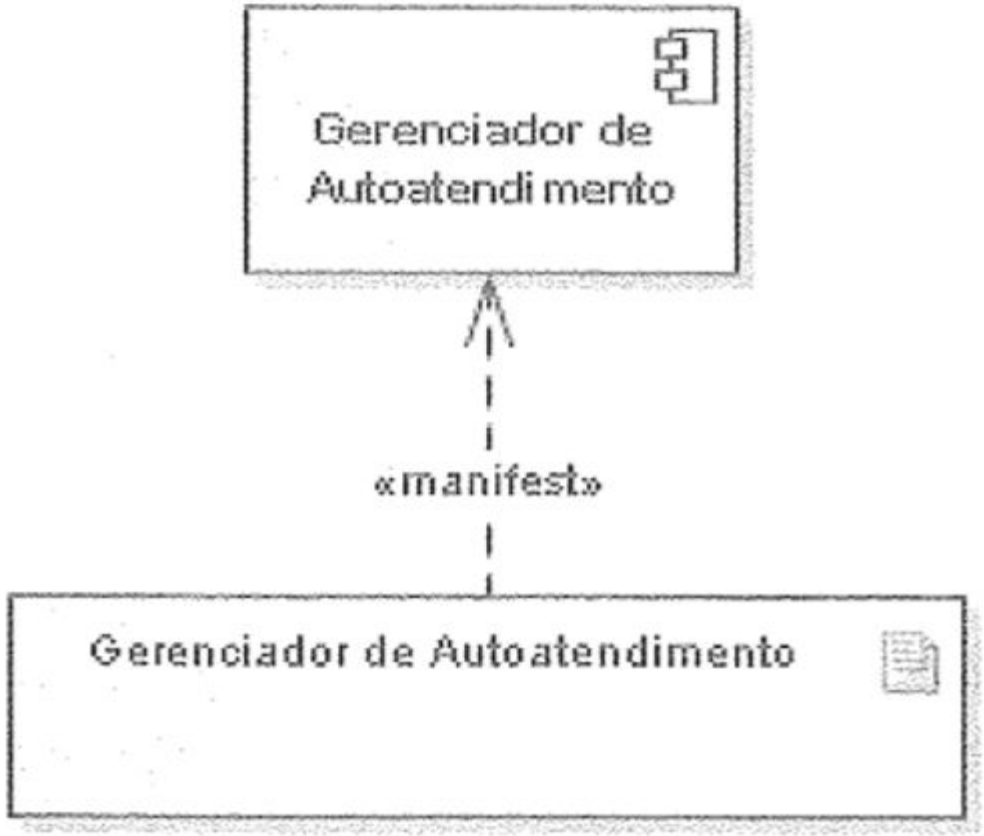




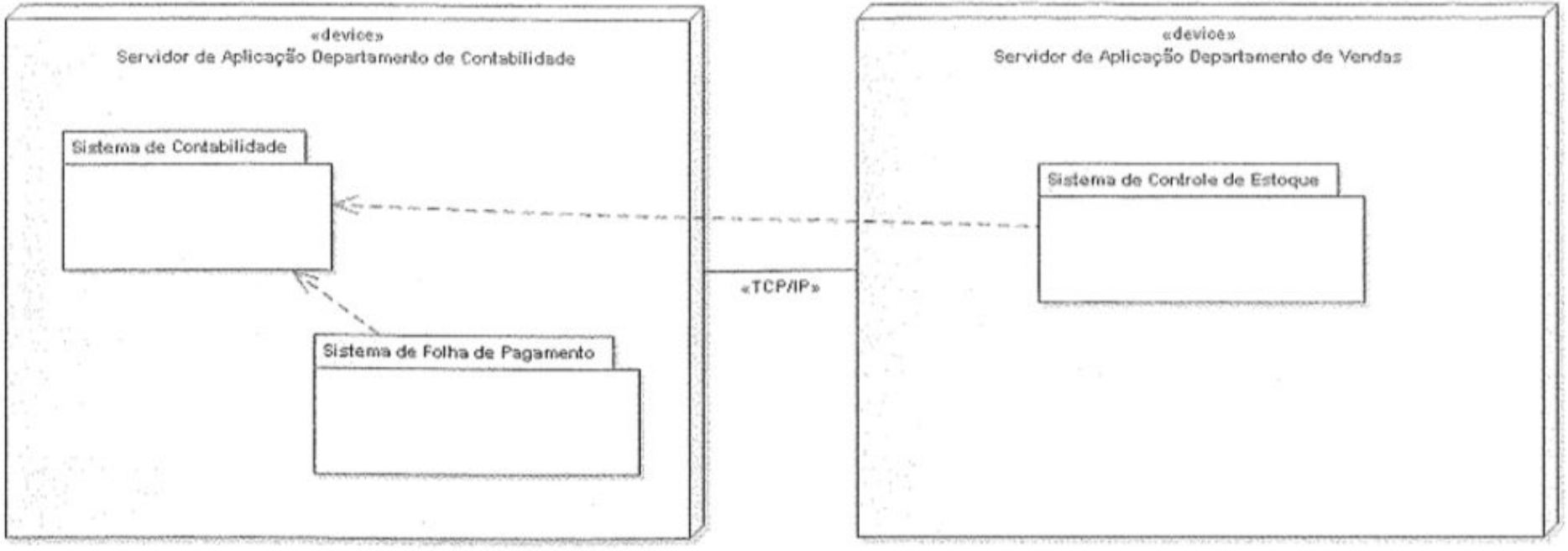
- O Estereótipo <<*deploy*>> na dependência indica que o artefato está implantado no nó

Artefato instanciado a partir de um Componente

- O estereótipo <<*manifest*>> na dependência indica que o artefato é uma manifestação do componente



Nós contendo Pacotes



Atividade de Fixação

- Explique o que é o diagrama de pacotes e o que ele permite modelar
- Explique o que é o diagrama de componentes e o que ele permite modelar
- Explique o que é o diagrama de implantação e o que ele permite modelar

Referências

BEZERRA, E. Princípio de Análise e Projeto de Sistemas com UML, Rio de Janeiro, Elsevier, 2007. **(Capítulo 11)**

LARMAN, C.; Utilizando UML e Padrões: uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento iterativo, Porto Alegre, Bookman, 2007. **(Parte V)**

GUEDES, Gilleanes T. A. UML 2: uma abordagem prática. 2. ed. São Paulo: Novatec, c2011. 484 p. ISBN 9788575222812 **(Capítulo 6, 12 e 13)**

Projeto de Software

Prof. Dr. Lesandro Ponciano

https://orcid.org/0000-0002-5724-0094