



Projeto da Arquitetura

Arquitetura do Sistema

A arquitetura de software de um sistema computacional é o conjunto de estruturas necessárias para discutir sobre o sistema, que inclui elementos de software, relações entre eles, e propriedades de ambos.

[SEI]

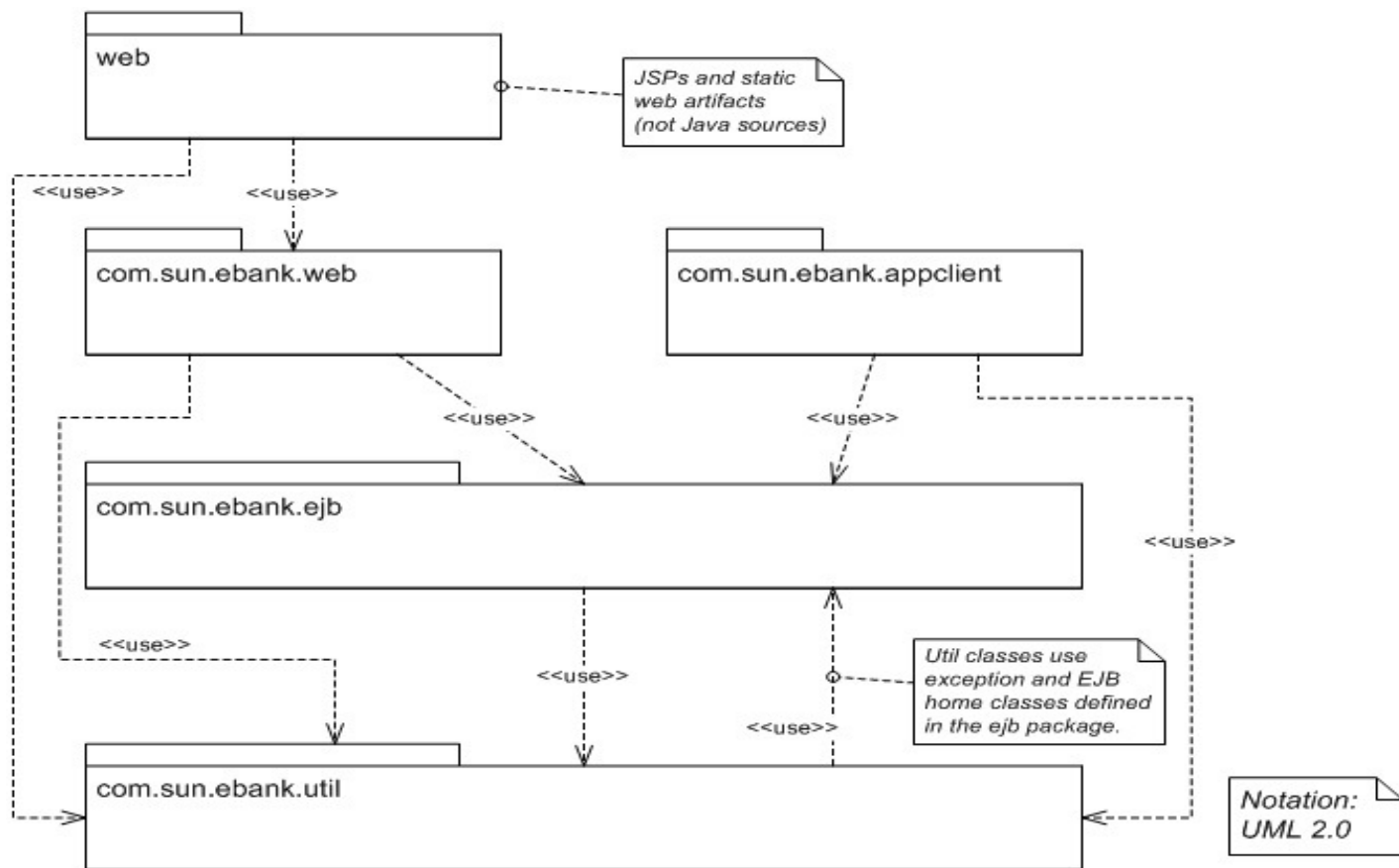
Uma visão é uma representação de um conjunto de elementos do sistema e os relacionamentos entre eles:

- Visão de Módulos
- Visão de Runtime
- Visão de Deployment
- Modelo de Dados

➔ O arquiteto deve escolher as visões que fazem mais sentido para o seu projeto.

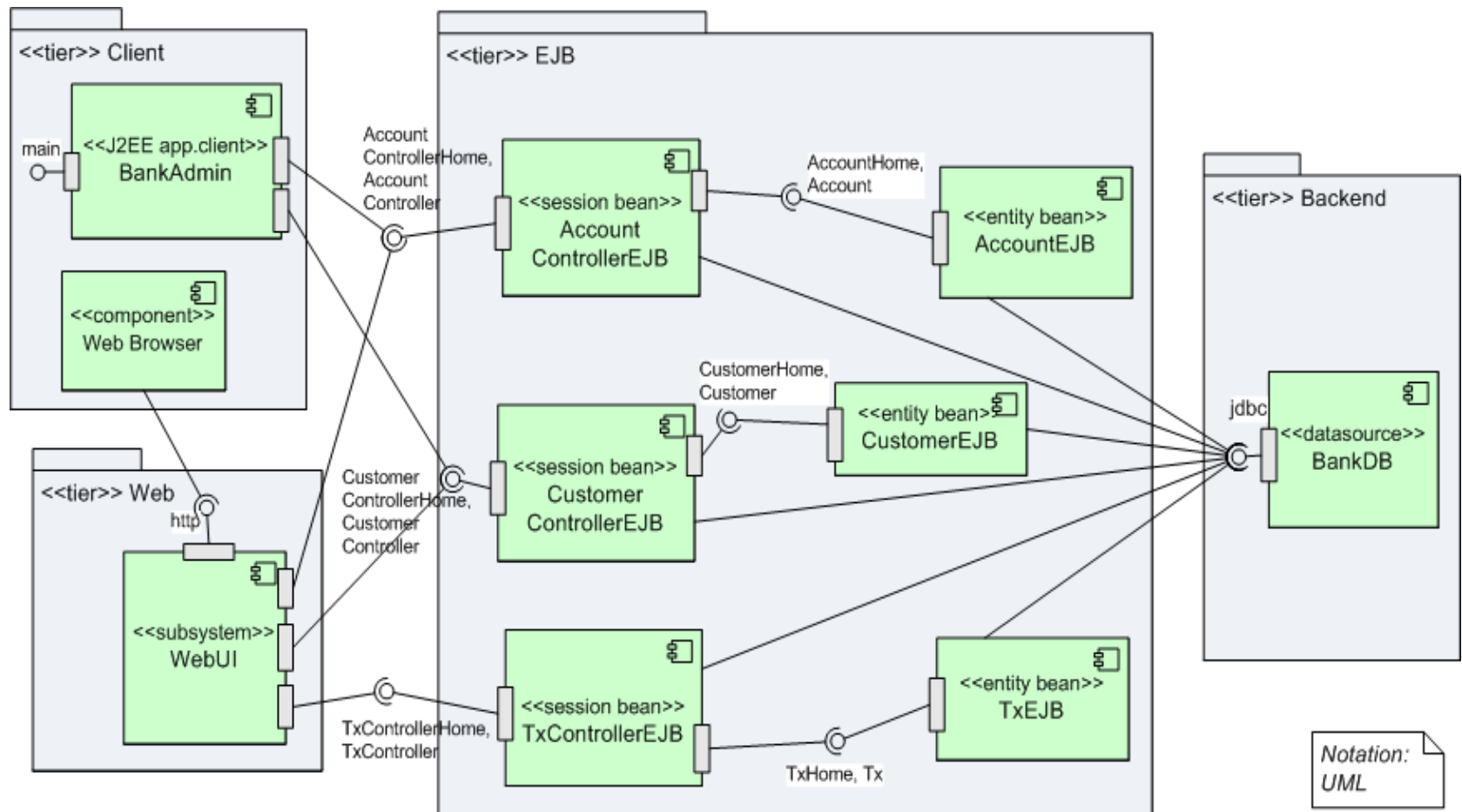
Visão de Módulos

Mostra a estrutura do sistema em termos de unidades de implementação (módulos).



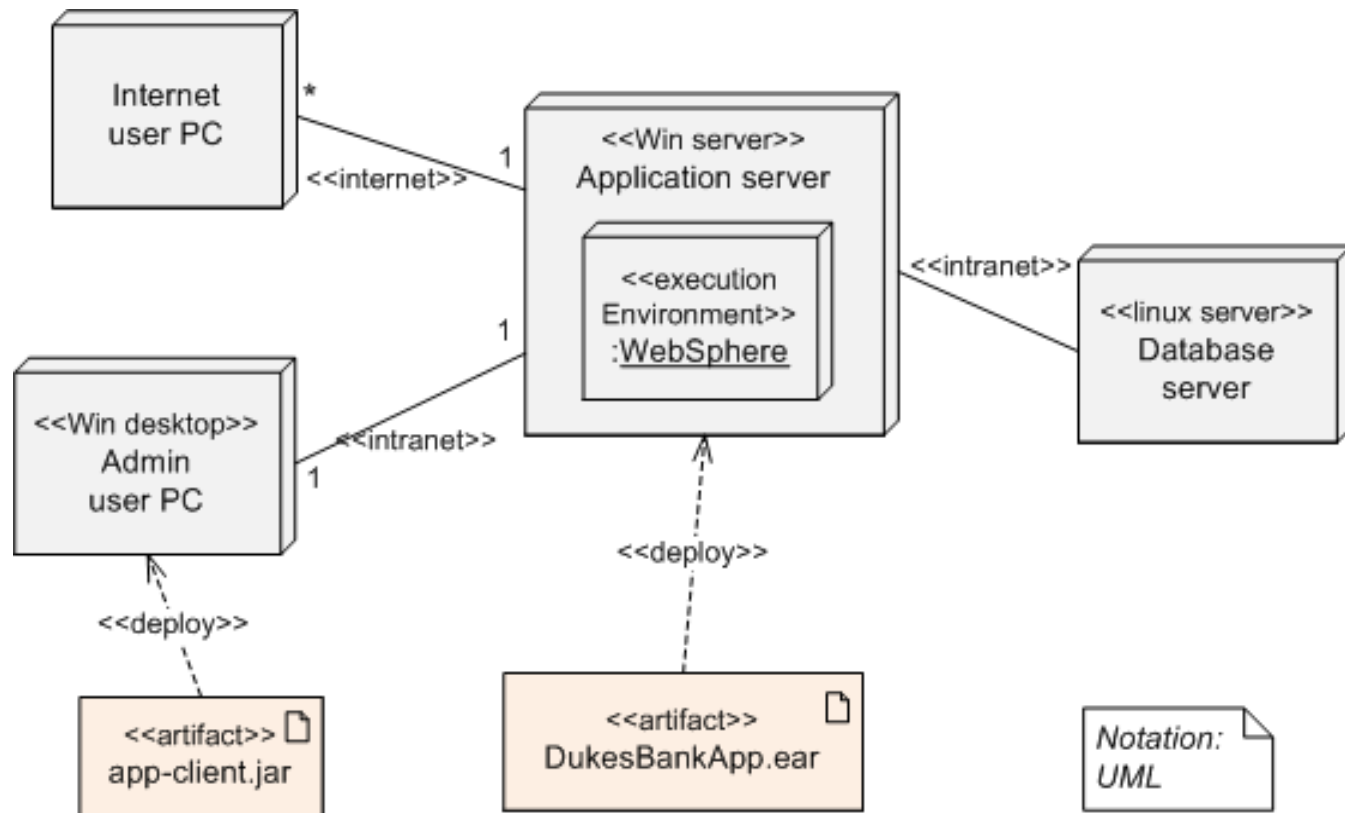
Visão de Runtime

Mostra a estrutura do sistema quando está sendo executado.



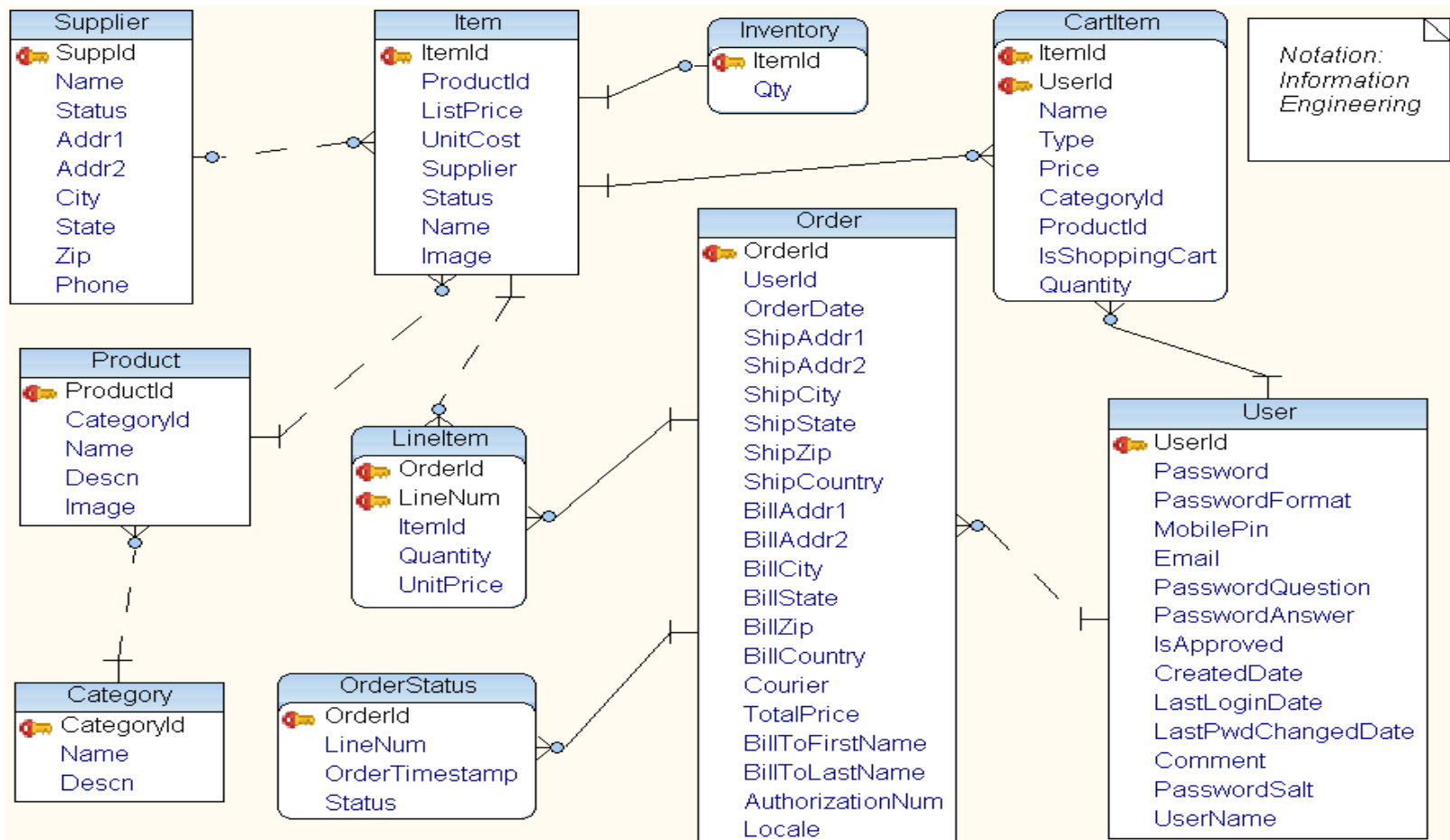
Visão de Deployment


Mostra a infraestrutura de hardware do ambiente de produção (nodos), incluindo os arquivos e componentes alocados aos nodos.



Modelo de Dados


Mostra a estrutura do repositório de dados (entidades persistentes)





Arquitetura do Sistema

➔ O arquiteto deve escolher as visões que fazem mais sentido para o seu projeto.



Projeto da Arquitetura

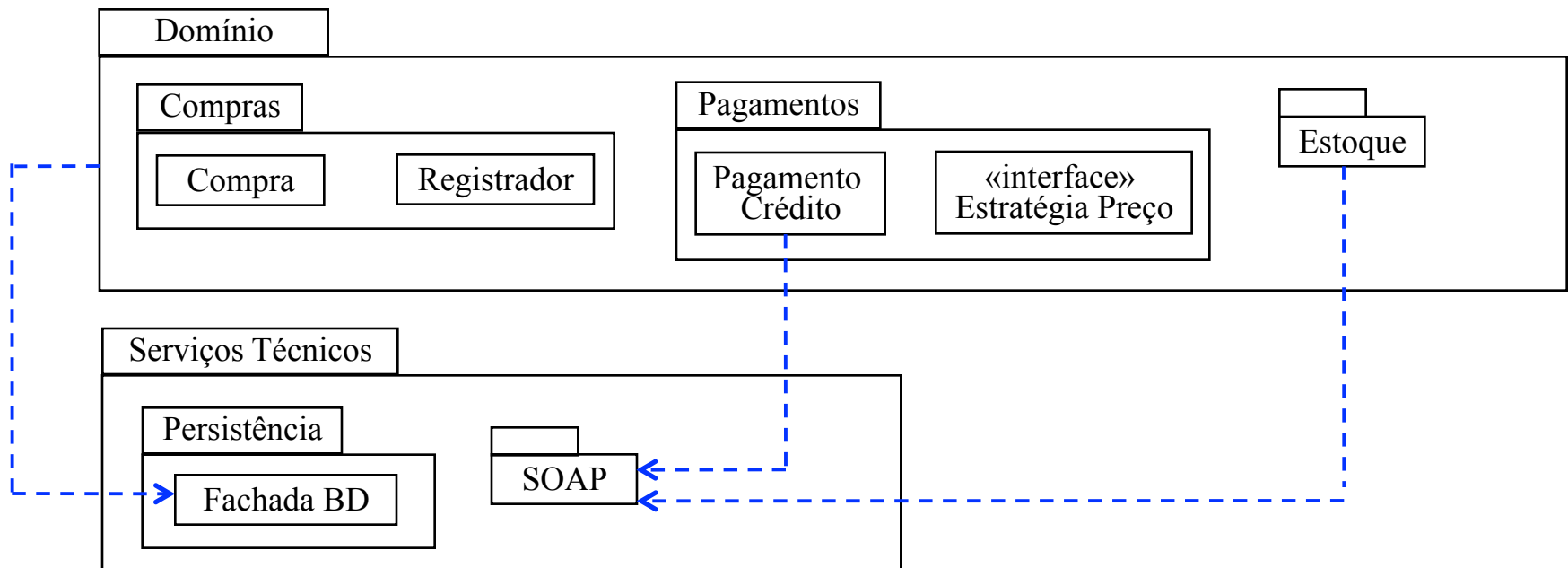
Detalhamento de duas visões:

- **Visão de Módulos (Lógica)**: descreve o sistema em termos da sua organização conceitual em camadas (layers), pacotes, classes, interfaces e subsistemas.
- **Visão de Deployment (Implantação)**: descreve o sistema em termos da alocação dos processos (subsistemas/componentes) em nodos de processamento.

Projeto Lógico da Arquitetura

- Processo de identificação dos sub-sistemas de um sistema e do estabelecimento da comunicação entre estes sub-sistemas.

↓ os subsistemas são representados através de camadas (layers)



Camadas (Layers) da Arquitetura

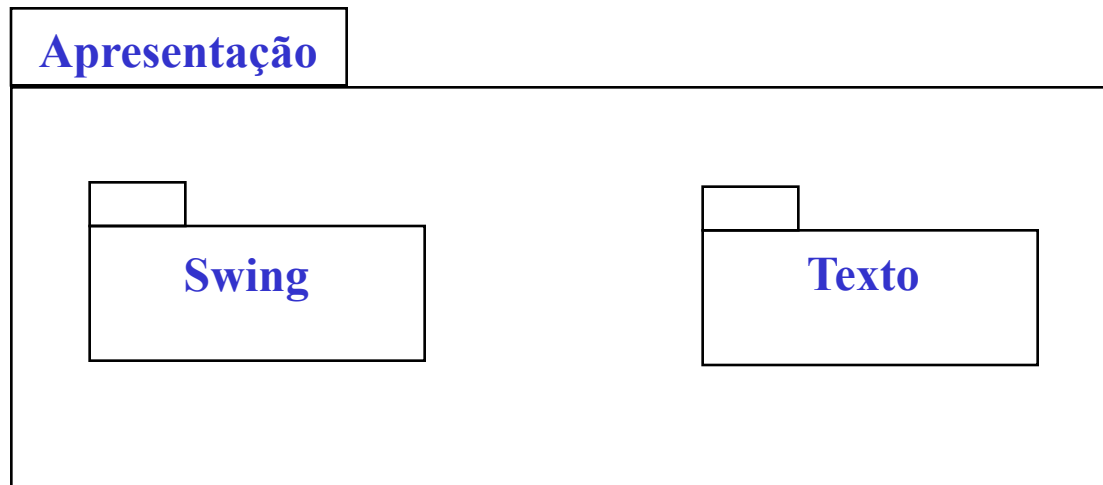
- ➔ Uma camada (layer) é um agrupamento de classes, pacotes ou subsistemas que tem uma responsabilidade coesiva de algum aspecto do sistema.
- ➔ A estrutura lógica de um sistema pode ser organizada em diferentes camadas de responsabilidades distintas e relacionadas:
 - as camadas mais baixas são serviços gerais e de baixo nível;
 - as camadas mais altas são serviços específicos da aplicação.
- ➔ Vantagem no uso de camadas:
 - isolar cada camada da outra. As camadas podem ser modificadas com pouco impacto nas outras.

Representação da Arquitetura Lógica

Os diagramas de pacote da UML são utilizados para representar a arquitetura lógica (camadas) do sistema.

↳ Uma camada pode ser modelada como um pacote UML.

Exemplo: a camada de Interface com Usuário pode ser modelada como um pacote chamado Apresentação.



Camadas

Geralmente um sistema OO inclui as seguintes camadas (layers):

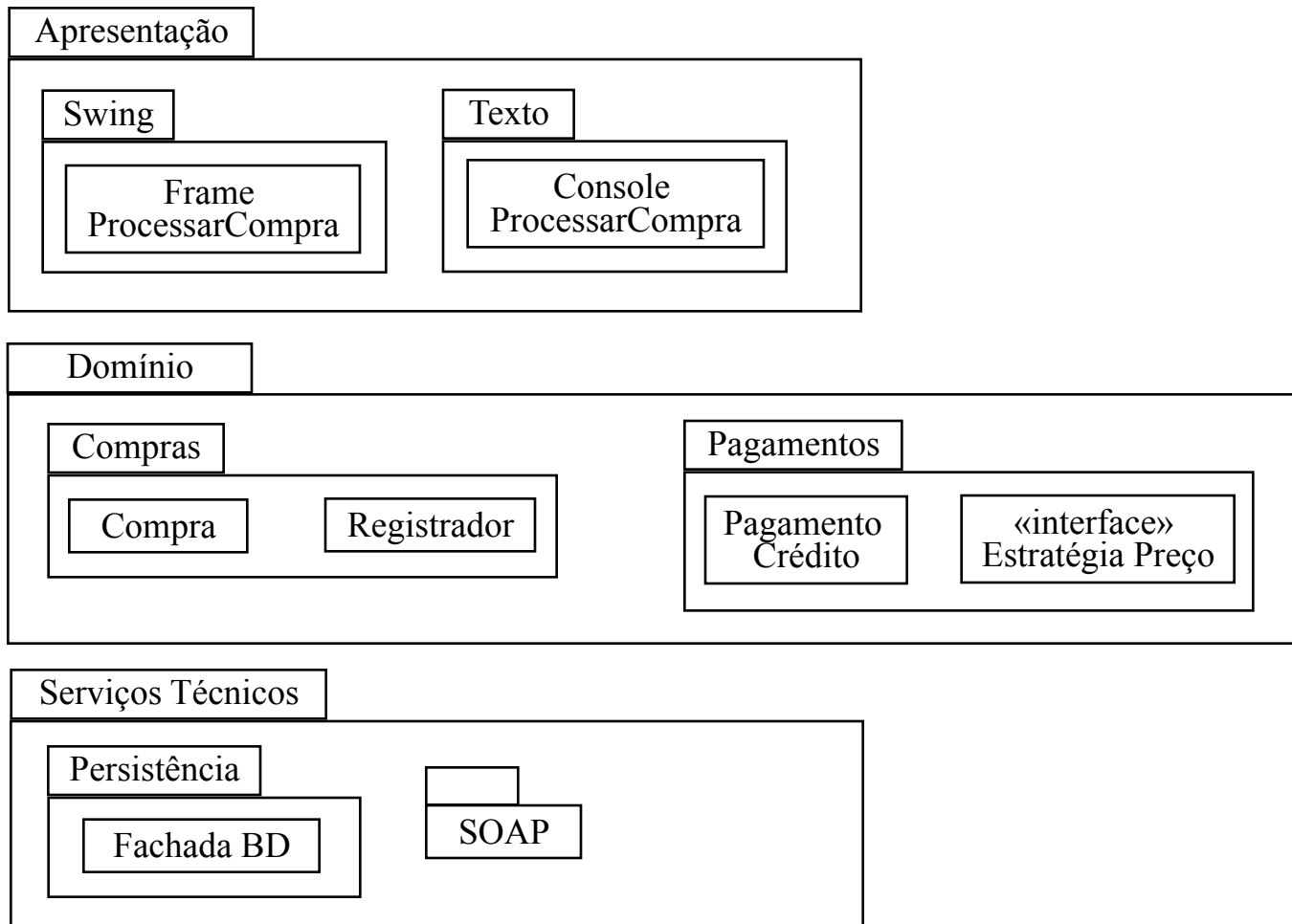
- Interface com o Usuário
- Lógica da Aplicação e Objetos do Domínio: objetos de software que representam conceitos do domínio.
- Serviços Técnicos: objetos e subsistemas de finalidade geral que fornecem suporte a serviços técnicos.

Exemplo: interfaceamento com uma base de dados.

Estes serviços são usualmente independentes da aplicação e reusáveis através de vários sistemas.

Arquitetura Lógica

Exemplo 1: Visão lógica parcial da aplicação NextGen



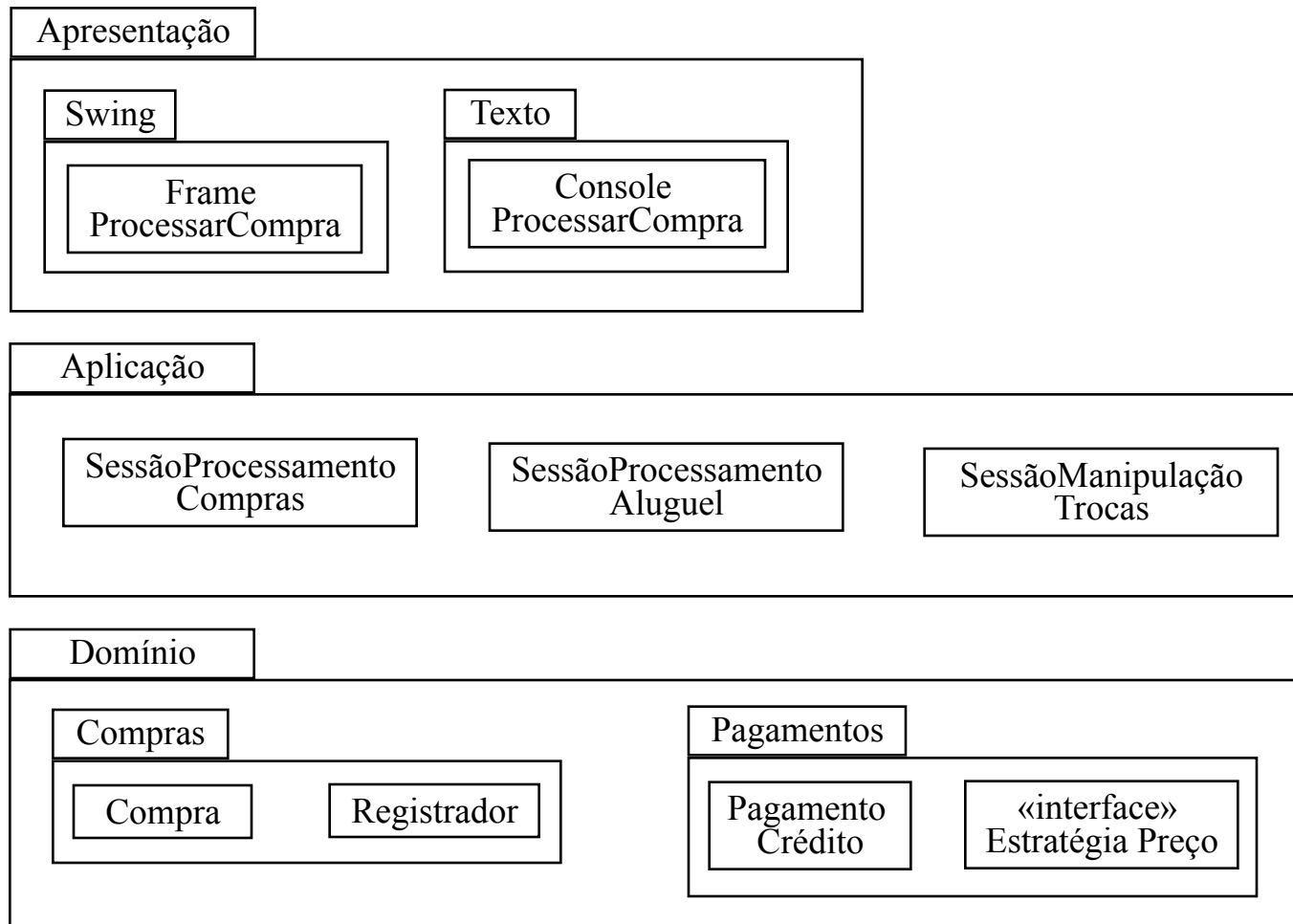
Camadas


A finalidade e o número de camadas variam nas aplicações:

- Apresentação ou Interface
- Aplicação ou Controle de Aplicação
Controlam os estados das sessões dos clientes.
- Domínio ou Negócio
Serviços do domínio.
- Infraestrutura do Negócio
Serviços gerais do negócio que podem ser usados em vários domínios.
Exemplo: Conversão de Moeda
- Serviços Técnicos
Exemplo: Persistência, Segurança
- Base (Foundation)
Exemplo: Estruturas de Dados, Rede, Banco de Dados

Arquitetura Lógica

Exemplo 2: Visão lógica parcial da aplicação NextGen





Dependência entre Camadas

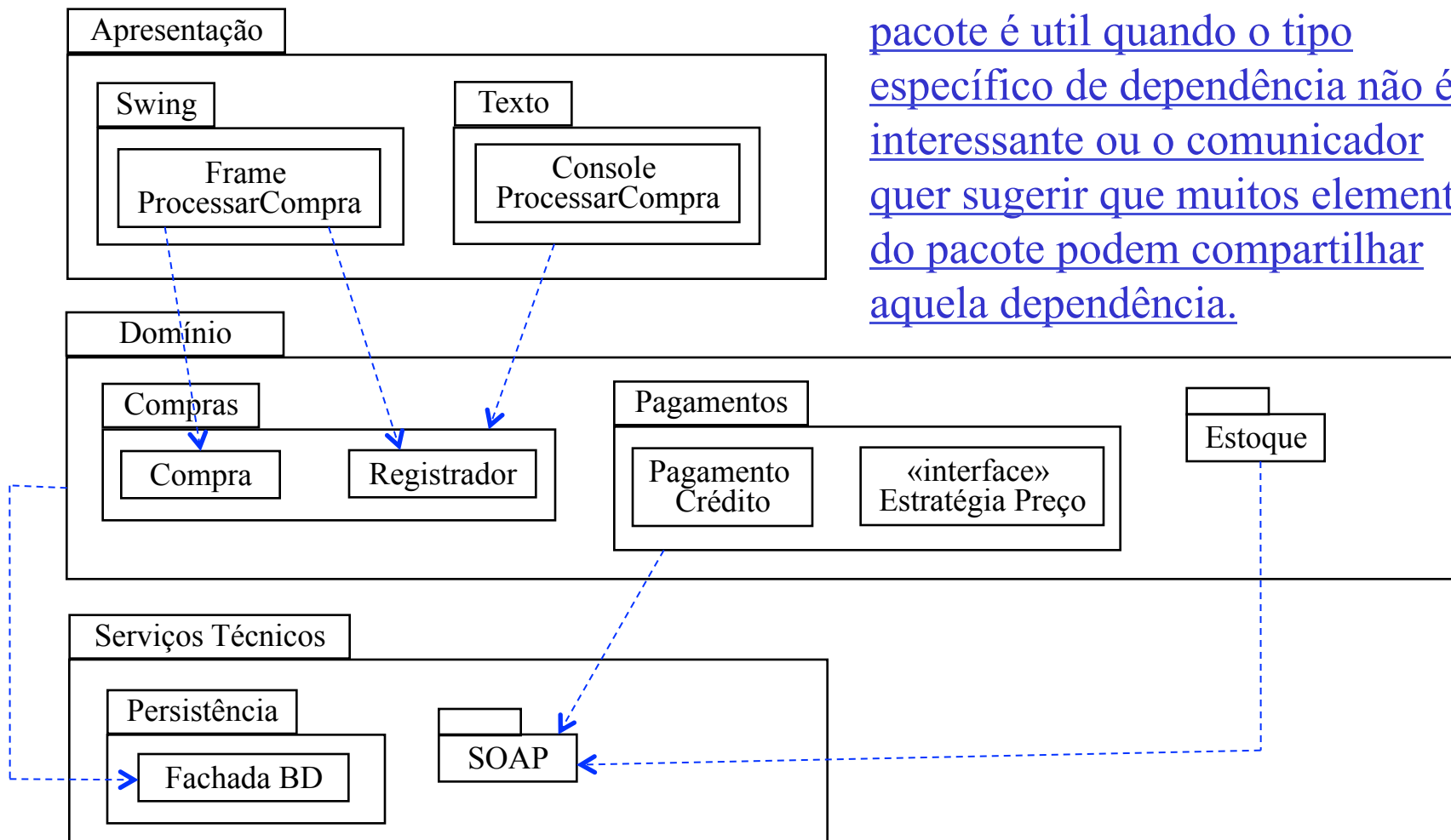
→ A colaboração e acoplamento é das camadas mais altas para as mais baixas.

Uma camada pode ser acessada por qualquer camada superior, e não somente pela camada imediatamente superior.

→ É interessante incluir um diagrama na visão lógica que mostre o acoplamento entre as camadas e pacotes.

Dependência entre Camadas

Exemplo:

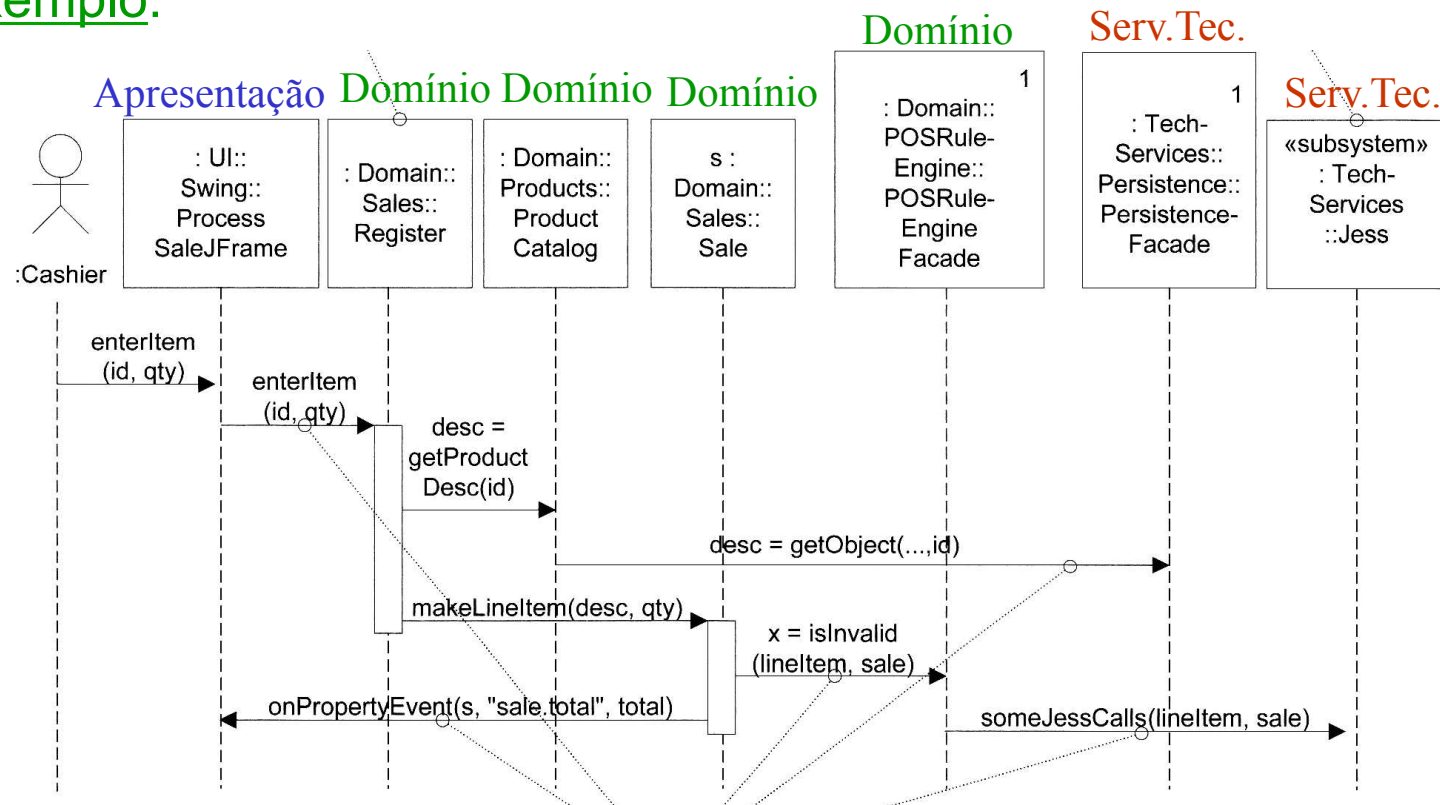


Observação: a seta saindo de um pacote é útil quando o tipo específico de dependência não é interessante ou o comunicador quer sugerir que muitos elementos do pacote podem compartilhar aquela dependência.

Camadas e Diagramas de Sequência

- ➡ Os diagramas de pacotes mostram as informações estáticas.
- ➡ Um diagrama de interação pode ser usado para entender como os objetos se conectam e se comunicam através das camadas.

Exemplo:



Arquitetura Lógica

Problemas que podem ser amenizados com o uso de uma arquitetura em camadas:

- A lógica da aplicação está misturada com a interface do usuário
 - ➔ não pode ser reusada com uma interface diferente, nem distribuída para um outro nodo.
- Serviços técnicos gerais e lógica de negócio são misturados com a lógica mais específica da aplicação
 - ➔ não podem ser reusados, nem distribuídos para outro nodo, nem substituídos por uma implementação diferente.
- Existe um alto acoplamento através de diferentes áreas de interesse
 - ➔ é difícil dividir o trabalho entre diferentes desenvolvedores.

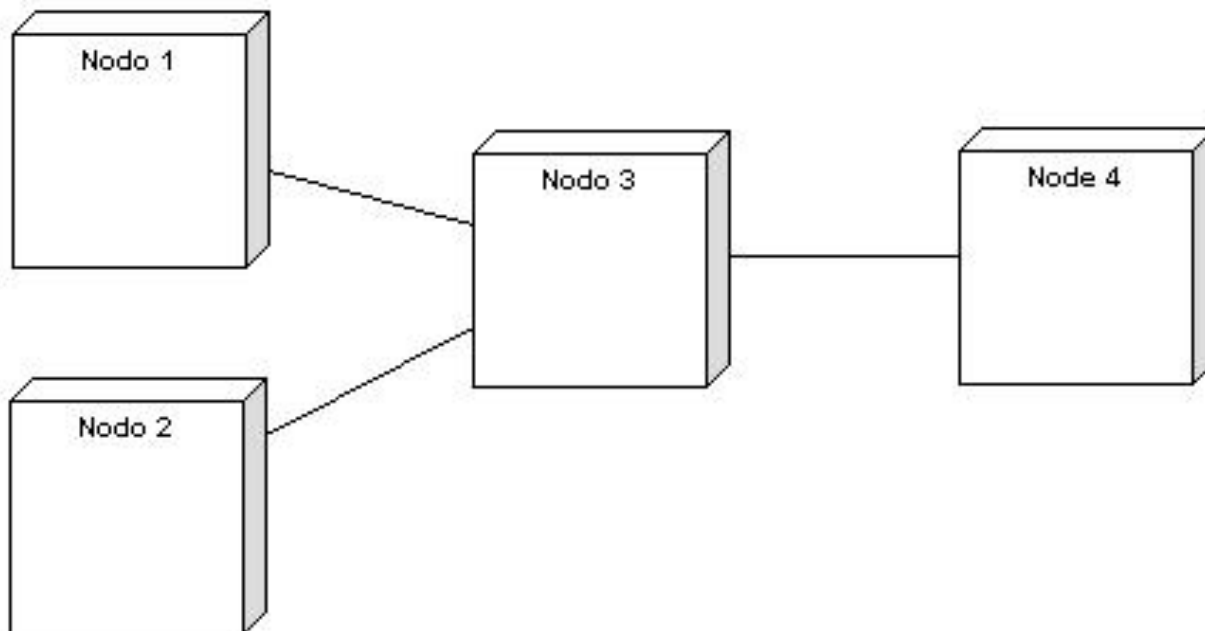
Arquitetura Lógica e de Deployment

- ➔ As camadas descrevem a organização conceitual dos elementos de projeto (pacotes ou subsistemas) em grupos, independente do seu empacotamento físico.
- ➔ Dependendo da plataforma, todas as camadas poderiam ser implantadas (*deployed*) dentro do mesmo processo ou espalhadas através de muitos processos.

Deployment

Arquitetura de Deployment: descreve o sistema em termos da alocação dos processos (subsistemas/componentes) em unidades de processamento.

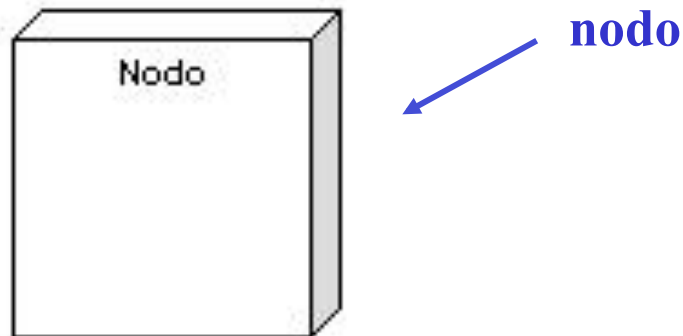
- É representada através de diagramas de deployment da UML.



Deployment - Tier x Nodo

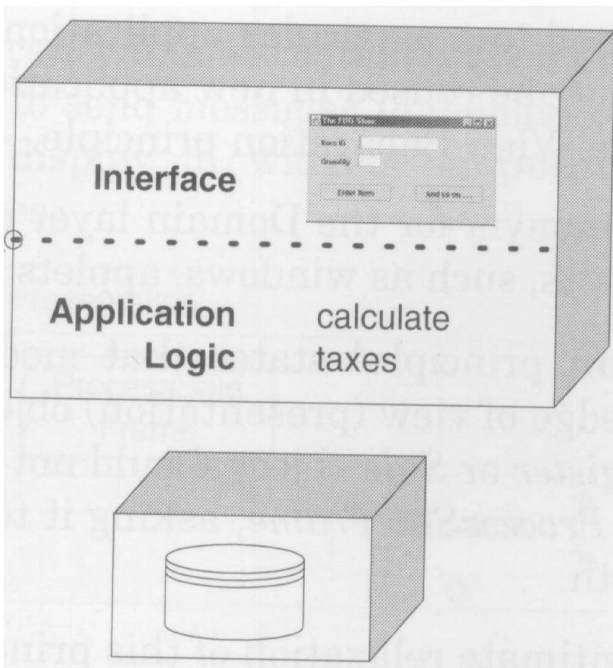
Tier: unidade independente de processamento (processo, máquina) capaz de se conectar com outros tiers.

Nodo: é um elemento físico que existe em tempo de execução e representa um recurso computacional, geralmente tendo pelo menos alguma memória e, frequentemente, capacidade de processamento (definição da UML).

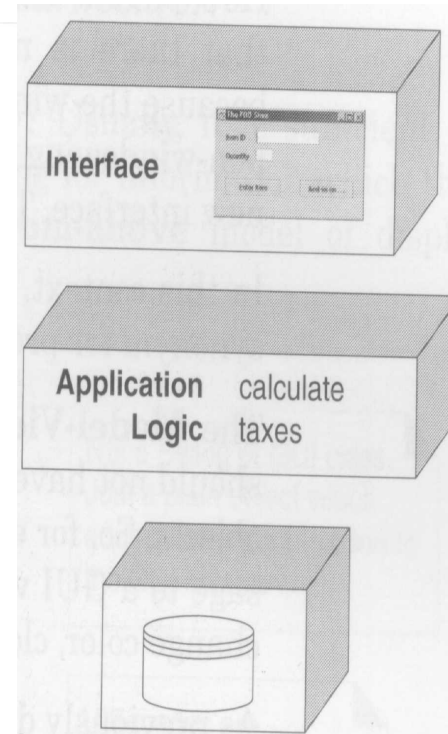


Arquitetura de Implantação

A alocação das camadas aos nodos pode variar.



3 camadas e 2 nodos



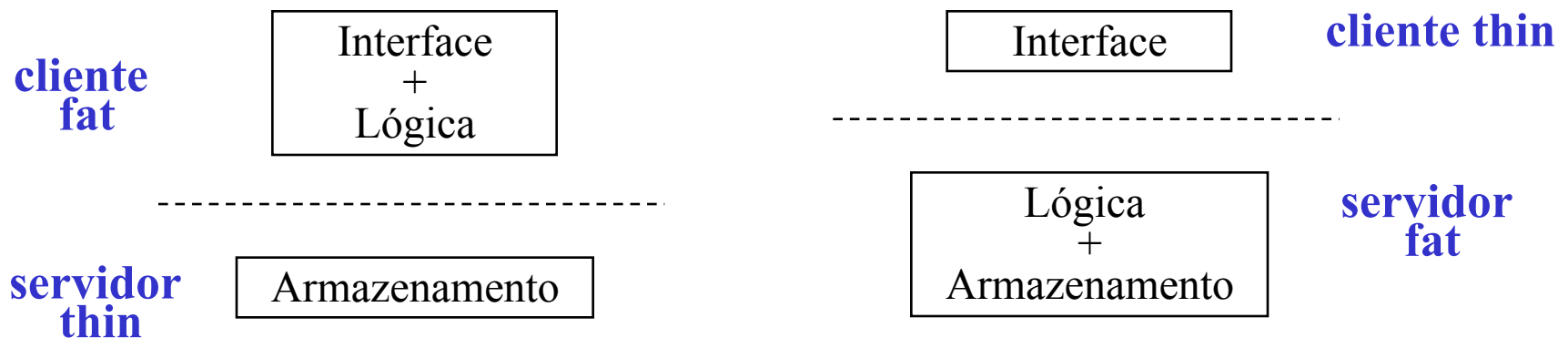
3 camadas e 3 nodos

Arquitetura Two-Tier

Inclui os seguintes tiers:

- Interface e lógica: a lógica da aplicação e a interface estão no mesmo tier (CLIENTE).
- Armazenamento: o mecanismo de armazenamento persistente está em um outro tier (SERVIDOR).

A lógica da aplicação também pode estar no servidor (stored procedures).



Two-Tier com uma Camada Apres./Dom.

A lógica da aplicação é colocada dentro das definições das janelas, que leem e escrevem diretamente na base de dados



Não define camadas diferentes de apresentação e de domínio



Desenvolvimento inicial rápido

Usado em aplicações que apresentam basicamente as seguintes operações sobre os dados: criação, recuperação, atualização e exclusão



Problemas de extensibilidade

Princípio da Separação Modelo-Visão

Modelo: objetos do domínio

Visão: objetos de interface com o usuário (UI)

➔ O princípio estabelece que os objetos do modelo (domínio) não devem ter conhecimento direto dos objetos de visão (UI).

Princípio da Separação Modelo-Visão

Diretrizes:

➔ Não conecte objetos não UI diretamente com objetos UI.


Exemplo: não deixe que um objeto Compra tenha uma referência para um objeto JFrame.

➡ Os objetos não UI podem ser utilizados em novas aplicações ou anexados a uma nova interface.

➔ Não coloque a lógica da aplicação nos métodos de um objeto UI.

Exemplo: o cálculo de uma taxa.

➡ Objetos UI devem somente inicializar elementos UI, receber eventos UI e delegar requisições da lógica da aplicação a objetos não UI.



Arquitetura Three-Tier

Inclui os seguintes tiers:

- Interface: janelas, formulários, etc.
- Lógica da Aplicação: tarefas e regras.
- Armazenamento: mecanismo de armazenamento persistente.

Característica principal:


- Separação da lógica da aplicação em um tier intermediário.

Em muitos casos, a parte principal da lógica da aplicação está no tier intermediário, enquanto algumas partes da lógica estão no tier de interface e de armazenamento.

Arquitetura Three-Tier

Exemplo: Aplicação WebMail

- Tier (browser): interface com o usuário, fornecida através da interpretação do HTML por um browser.
- Tier (webmail): web server que dispara processos (cgi, servlet, etc) que se conectam com o servidor de mail.
- Tier (servidor de mail)




Arquitetura com mais de 3 Tiers

Algumas aplicações podem ser melhor projetadas usando mais de 3 tiers.

Exemplo: aplicações com várias fontes de dados integradas

- Tier: front end.
- Tier: servidor da aplicação que executa consultas baseadas na visão unificada dos dados.
- Tier: servidor que unifica a visão destes dados.
- Tier: repositórios de dados individuais.



Projeto da Arquitetura

Qual a diferença entre arquitetura e design?

Arquitetura é design, mas nem todo o design é arquitetural.

O design pode ser dividido em:

- design arquitetural
- design não-arquitetural

O projetista define a linha entre os dois.

Scott Ambler disse em uma palestra que essa pergunta não é muito importante, o importante é o projetista saber o quanto de design é necessário.