# Jakarta (Enterprise) Java Beans
# Software Coorporativo - Introdução

Prof. Dr. Alexandre L'Erario


# Pré-requisitos

- Conhecimento em Java SE
  - Conceitos de Orientação a Objetos
  - Classes, Interfaces, Relacionamento
- Programação distribuída em Java
  - RMI
- Programação Web:
  - JSF

# Sumário

- Conceitos de Software Corporativo
- Arquitetura de Software

# O que é software corporativo

- Software que resolve problemas da corporação e não apenas de um departamento
- Customizações são necessárias de organização para organização
- É comumente desenvolvido sobre em uma arquitetura comum
- Projetado para atender muitos clientes, de muitas maneiras, de muitos dispositivos
- Projetado para otimizar performance
  - Threads, por exemplo
- Comumente implementados sobre um *Middleware*

# Desafio do software corporativo

- **Somente** um problema:
  - Manter o software **usável**
  - Volatilidade constante de:
    - Requisitos funcionais
    - Requisitos não funcionais

- **Muitas** soluções:
  - Desenvolver novas funcionalidades
  - Adotar novas técnicas/tecnologias

# Vejamos algumas consequências

- Integrações cada vez mais complexas
- Operações cada vez mais complexas
- Métricas complexas (custo, prazo)
- Treinamento de usuários e desenvolvedores
- Controle de foco e escopo
- Personalização do software para cada cliente

# As soluções para estes problemas...

- Muitos dos requisitos não funcionais devem ser incorporados na arquitetura
- Requisitos funcionais mais significativos devem ser incorporados na arquitetura
- Uma arquitetura bem estruturada pode resolver muitos problemas
  - Por isso, devemos conhecer arquitetura

# Arquitetura de software

- Não há padrão universalmente aceito para o termo
- Definições Modernas:
  - *"Conjunto de estruturas de Software"*
  - *"Estrutura ou estruturas do sistema, as quais compreendem elementos de software, propriedades externamente visíveis de tais elementos (serviços providos, características de performance, tolerância à falhas, etc.) e as relações entre eles" - Bass, Clement e Kazman*
  - *"Arquitetura de Software pode ser definida como a disciplina de análise, definição e composição do Design Arquitetural" – Sommerville*
  - ***"Arquitetura é a prática recomendada como a organização fundamental de um sistema, encorpada em seus componentes, as relações entre eles e com o ambiente e os princípios que determinam seu design e evolução" – IEEE 1471***

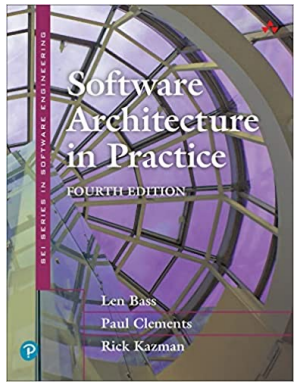
# ABNT/ISO 12207:2021

- Visão da arquitetura:
  - Produto de trabalho que expressa a arquitetura de um sistema a partir da perspectiva de interesses específicos do sistema.

- Processo de Definição de Arquitetura
  - Gerar alternativas de arquitetura do sistema. Selecionar uma ou mais alternativas que abranjam os interesses dos *stakeholders* e os requisitos do sistema e expressar em um conjunto de visões consistentes.
  - Usada para que haja um entendimento negociado do problema a ser resolvido e para que uma solução satisfatória seja identificada.
  - Os resultados do Processo de Definição de Arquitetura são amplamente utilizados ao longo dos processos do ciclo de vida.
  - A definição da Arquitetura pode ser aplicada em vários níveis de abstração, destacando os detalhes relevantes que são necessários para as decisões neste nível.

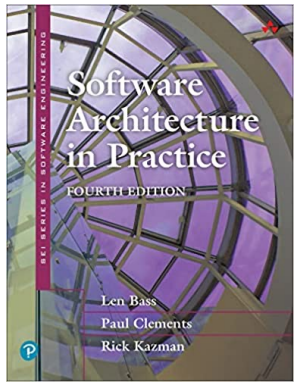
# Por que a arquitetura de software é importante?

- Inibindo ou Habilitando os Atributos de Qualidade de um Sistema
- Raciocínio e gerenciamento de mudanças
- Predição das qualidades do sistema
- Comunicação entre as partes interessadas
- Decisões de Projeto Antecipadas
- Restrições na implementação
- Influências na estrutura organizacional
- Habilitando o Desenvolvimento Incremental
- Estimativas de custo e cronograma
- Modelo Transferível e Reutilizável
- A Arquitetura Permite a Incorporação de Desenvolvidos Independentemente Elementos
- Restringindo o vocabulário de alternativas de design
- Uma base para o treinamento



# Atributos de qualidade

- Disponibilidade
- Implementabilidade
- Eficiência energética
- Integrabilidade
- Modificabilidade
- Desempenho
- Safety
- Security
- Testabilidade
- Usabilidade
- Outros Atributos de Qualidade



# Conjunto de estruturas de Software

1. Component-and-connector structures
2. Module structures
3. Allocation structures

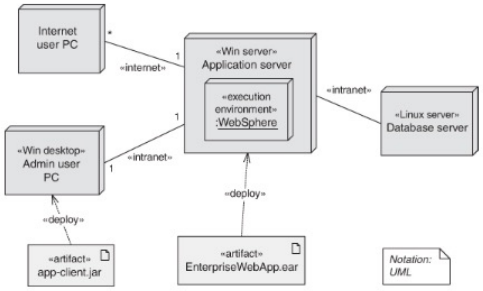



**Figure 1.10** *Deployment structure*

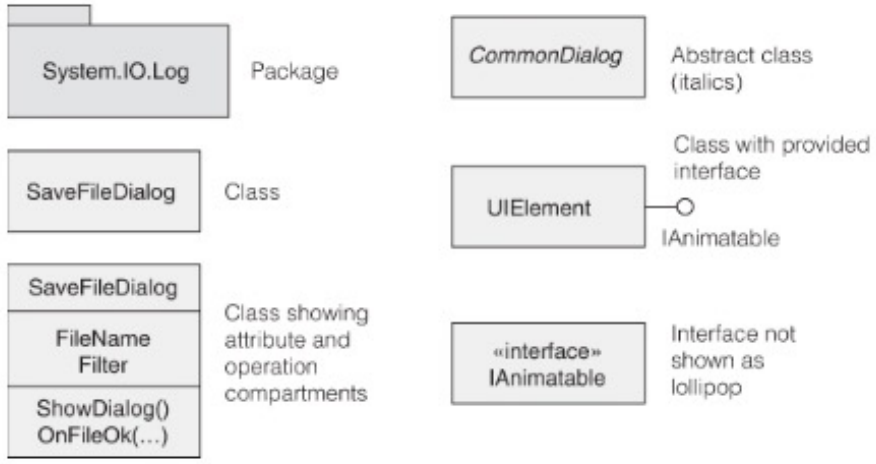



**Figure 1.3** *Module elements in UML*

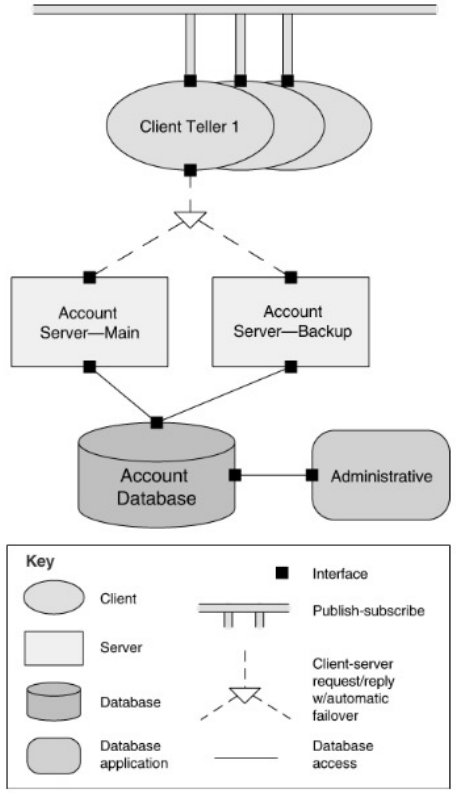



**Figure 1.2** *A component-and-connector structure*

# Soluções arquiteturais

- Software Interfaces
- Virtualization
- Cloud and Distributed Computing
- Mobile Systems

# Considerações finais

- Arquiteto de software deve conhecer:
  - Ambientes técnicos
  - Desenvolvedores / Gerentes
  - Clientes
- Habilidades do arquiteto:
  - Zelar pela padronização e comunicação
  - Identificar estruturas para resolver problemas
  - Minimizar a diversidade de estruturas
  - Maximizar o reuso de estruturas