# Engenharia de Software I

**Profª. Me. Cynara Leão Garcia**

cynara.garcia@unicesumar.edu.br

# Reutilização de Software

# Reutilização de Software

- Abordagem de desenvolvimento com o objetivo de maximizar o uso de software pré-existente

- Nos últimos 20 anos, muitas técnicas foram desenvolvidas para apoiar o reuso

    - Bibliotecas, objetos, componentes, etc.

- O movimento open source cria uma enorme base de código disponível

# Tipos de Reutilização

- Objetos e Funções
  - Tipo mais comum de reutilização
  - Ocorre nos últimos 40 anos

- Componentes
  - Reuso de média granularidade. Exemplo, componente arquitetural ou sub-sistema

- Sistema
  - Um sistema pode ser reusado por incorporação à outro sistema
  - Pode ser necessário customização

# Vantagens de Reutilização

- Redução de tempo e custos
  - O sistema pode ser entregue em menor prazo, que reduz os custos
- Maior confiabilidade do produto
  - O software reusado já foi testado antes
- Uso eficaz de especialistas
  - Especialistas compartilham o conhecimento
- Adequação aos padrões
  - Padrões de interface podem ser componentes reusáveis

# Potenciais Problemas

- Custo de Manutenção

  - Componentes reusados podem se tornar incompatíveis em versões futuras

- Falta de Apoio de Ferramenta

  - Ambientes de desenvolvimento podem não estar preparados para reutilização

- É caro manter uma biblioteca

  - É difícil encontrar e entender o software que se pretende reusar

# Planejamento para Reutilização

- Reutilização não ocorre por acaso
  - Ele deve ser planejado e incentivado em toda a organização

- Muitas empresas desenvolvem sistemas em um mesmo domínio
  - Surgem situações potenciais para reutilização

UniCesumar

# Fatores do Planejamento

- Alguns fatores a considerar no planejamento de reutilização
  - Cronograma de desenvolvimento
  - Ciclo de vida do software
  - Conhecimento e experiência da equipe
  - Domínio da aplicação

# Cronograma e Ciclo de Vida

- Cronograma de Desenvolvimento

  - Se o cronograma de entrega é apertado, reusar pode agilizar a entrega do produto

- Ciclo de Vida do Software

  - Reusar pode ser um problema em sistemas que sofrem manutenções frequentes

  - Componentes de terceiros (código proprietário) dificultam a manutenção

# Equipe e Domínio

- Conhecimento e experiência da equipe na abordagem de reutilização

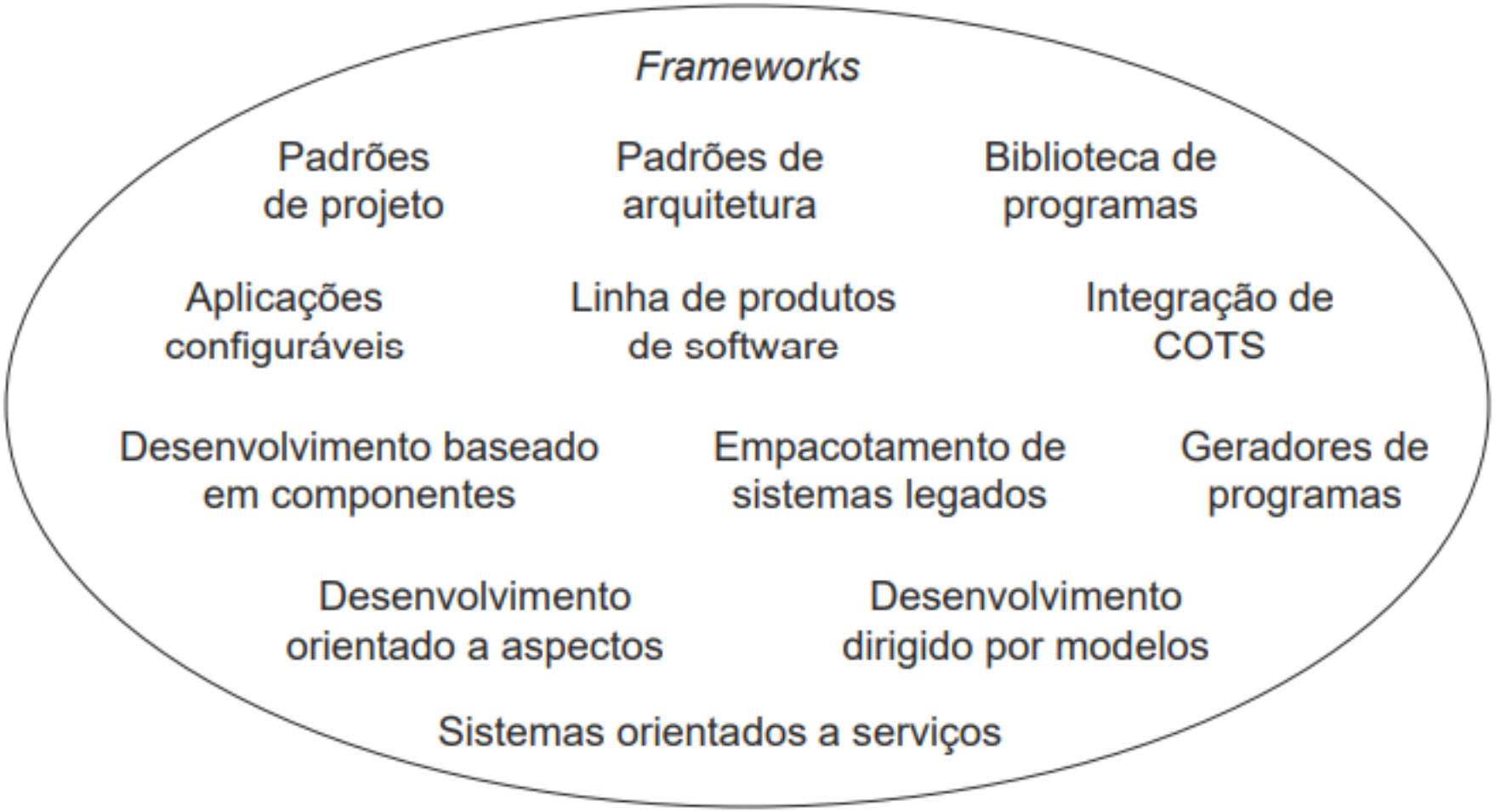
  - Muitas abordagens são difíceis de serem usadas e requerem experiência

- Domínio da aplicação

  - Em alguns domínios, é fácil encontrar componentes e bibliotecas para reusar

  - Em outros domínios, é mais complicado

# Panorama de Reutilização



# Técnicas (1 de 3)

- Padrões de projeto e de arquitetura

  - Padrões são soluções genéricas para problemas recorrentes

- Frameworks de aplicação

  - Coleção de classes abstratas e concretas que criam a estrutura de uma aplicação

- Componentes

  - Subsistemas que são integrados para criação da aplicação

# Técnicas (2 de 3)

- Empacotamento de sistemas legados

  - Sistemas legados são empacotados pela definição de interfaces de acessos

- Sistemas orientados a serviços

  - Sistemas desenvolvidos pela criação de serviços compartilhados

- Linhas de produtos de software

  - Família de aplicações desenvolvidas em torno de uma arquitetura comum

# Técnicas (3 de 3)

- Bibliotecas de programas

  - Classes e funções que implementam abstrações comumente usadas

- Desenvolvimento dirigido por modelos

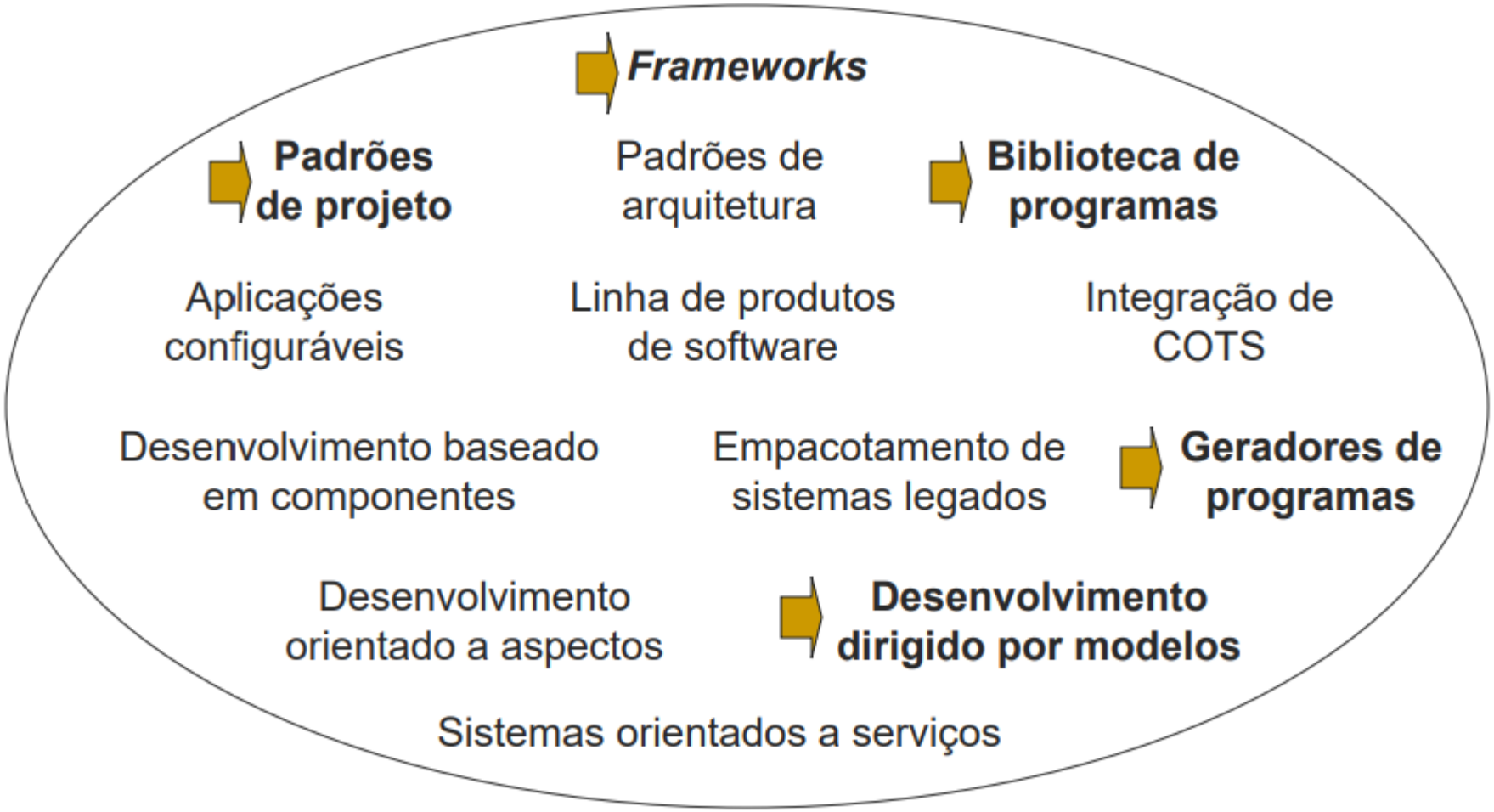
  - O código é gerado a partir de modelos de domínio e modelos de implementação

- Desenvolvimento orientado a aspectos

  - Técnica avançada de modularização de código para apoiar a reutilização


# Técnicas para Reutilização de Software

# Panorama de Reutilização



# Bibliotecas de Software

- Bibliotecas implementam serviços que podem ser usados por programas
  - É uma forma comum de reutilização

- Disponibiliza funcionalidades comuns a diferentes tipos de sistemas
  - Converter informação entre formatos conhecidos (e.g., string para inteiro)
  - Acesso a recursos, arquivos, BD, etc.
  - Tipos abstratos de dados: fila, pilha, lista...

# Uso de Biblioteca em Java

```
import java.util.Vector;

public class Customer {

  String name;
  Vector phoneNumbers = new Vector();

  void removePhoneNumber(String c){
    phoneNumbers.removeElement(c);
  }

  void addPhoneNumber(String c){
    phoneNumbers.addElement(c);
  }

  ...
}
```

UniCesumar

# Framework

- Frameworks são aplicações incompletas

  - São formados por interfaces, classes abstratas e classes concretas (OO)

- O conjunto de classes e interfaces formam uma estrutura genérica

- Um sistema é implementado pela adição de componentes para preencher lacunas

  - Por exemplo, pela implementação das classes abstratas do framework

# Tipos de Frameworks

- Frameworks de infra-estrutura

  - Apoiam a criação de infra-estruturas de sistemas, tais como comunicações, interfaces de usuário e compiladores

- Frameworks de integração

  - Apoiam a comunicação e a troca de informações de componentes

- Frameworks de aplicações

  - Apoiam o desenvolvimento de um tipo de aplicações (e.g., aplicações Web)

# Principal Problema

- Framework é normalmente uma entidade grande e complexa
  - Pode levar um longo tempo para entendê-lo e usá-lo efetivamente
  - O desenvolvedor pode querer apenas uma funcionalidade simples

# Padrões de Projeto

UniCesumar

# Padrões de Projeto

- Um padrão é uma descrição do problema e a essência da sua solução

- Documenta boas soluções para problemas recorrentes

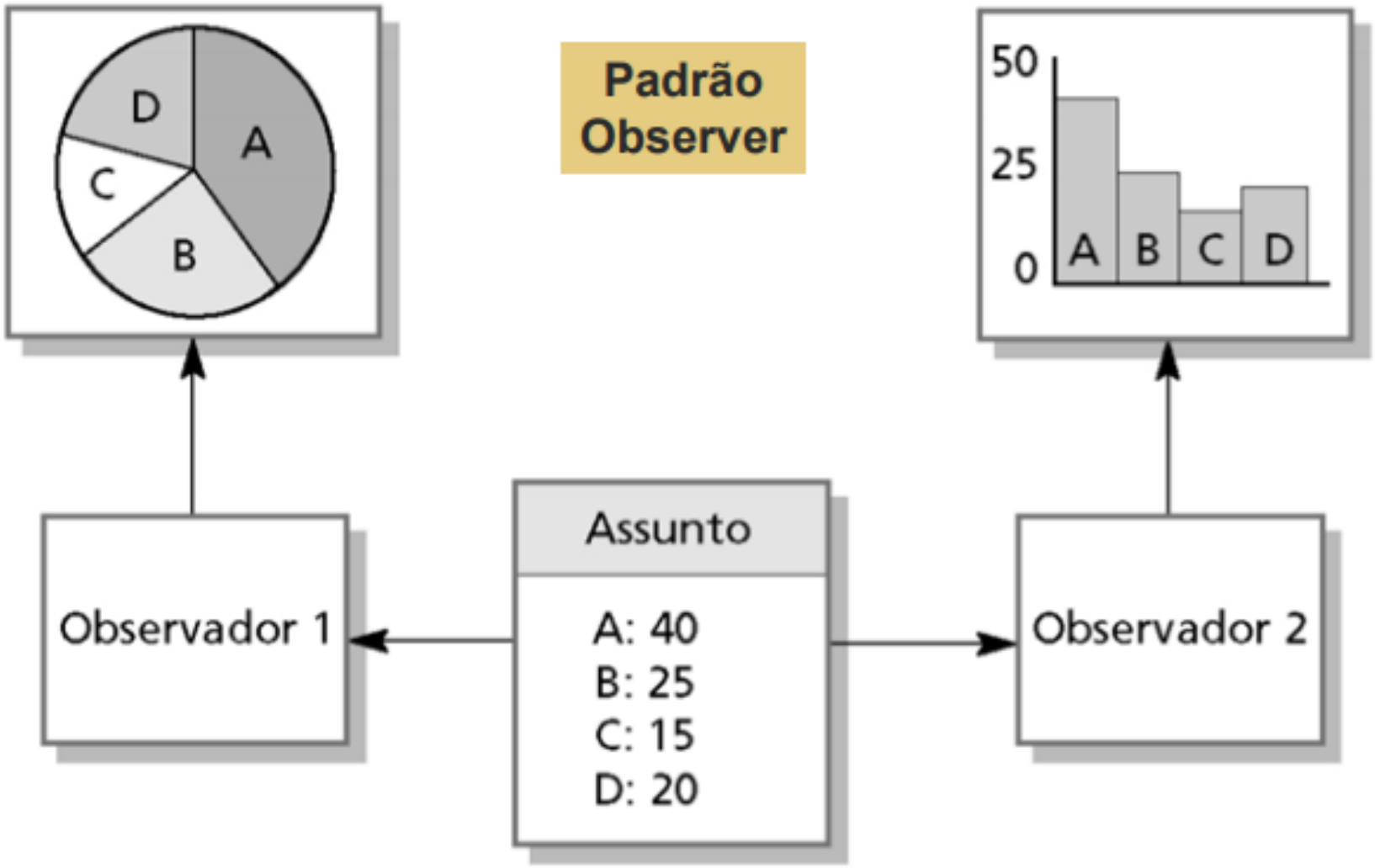
  - Permite a reutilização de conhecimento anterior documentados em boas práticas

- Deve ser suficientemente abstrato para ser reusado em aplicações diferentes

# Elementos de um Padrão

- Nome
  - Um identificador significativo para o padrão
- Descrição do problema
- Descrição da solução
  - Um template de solução que pode ser instanciado em maneiras diferentes
- Consequências
  - Os resultados e compromissos de aplicação do padrão

# Exemplo de Problema




UniCesumar

# Padrão Observer

- Nome
  - Observer
- Descrição do problema
  - Separa o objeto de suas formas de apresentação
- Descrição da solução (próximo slide)
- Consequências
  - Otimizações para melhorar a atualização da apresentação

# Linha de Produtos de Software

# Linha de Produtos de Software

- LPS é um conjunto de aplicações definidas sobre uma arquitetura comum e que compartilham componentes

SEI

- Engenharia de LPS é um processo de desenvolvimento de aplicações de software usando uma arquitetura comum e customizações

Klaus Pohl

UniCesumar

# Propriedades de uma LPS

- LPS favorece a reutilização em larga escala
- É geralmente criada a partir de várias aplicações desenvolvidas sobre o mesmo domínio
- Usa outras técnicas de reutilização
    - Frameworks, componentes, padrões, etc.

# Reuso vs. Customização

# Melhoria da Qualidade

- Os artefatos de uma LPS são revisados e testados em muitos produtos

  - Estes artefatos tendem a atingir elevado grau de confiabilidade (poucas falhas)

- Artefatos confiáveis elevam a qualidade de todos os produtos da LPS

# Modelo de Característica - Conceito

- Característica (*feature*) é uma propriedade importante e observável do sistema

    - Um modelo de característica permite expressar a configurabilidade da LPS

- Uma característica deve ter um nome conciso

    - Facilita a comunicação entre desenvolvedores

# Elementos de uma LPS

- Elementos mandatórios

  - Aqueles que são encontrados em todos os membros da linha de produto

- Elementos variáveis

  - **Opcionais:** um produto pode ou não contê-los

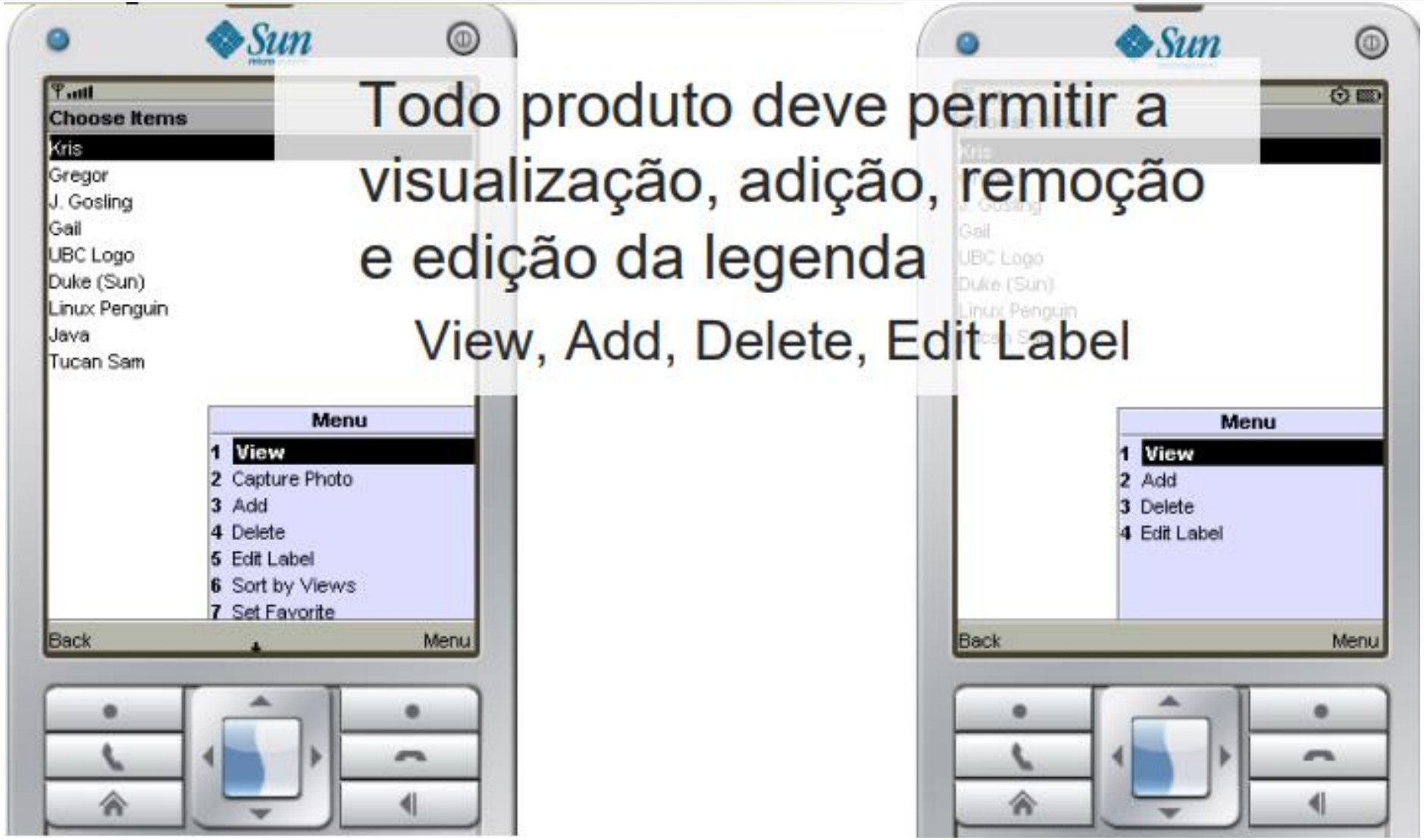
  - **Alternativos:** um produto deve conter uma das alternativas

# Exemplo de LPS

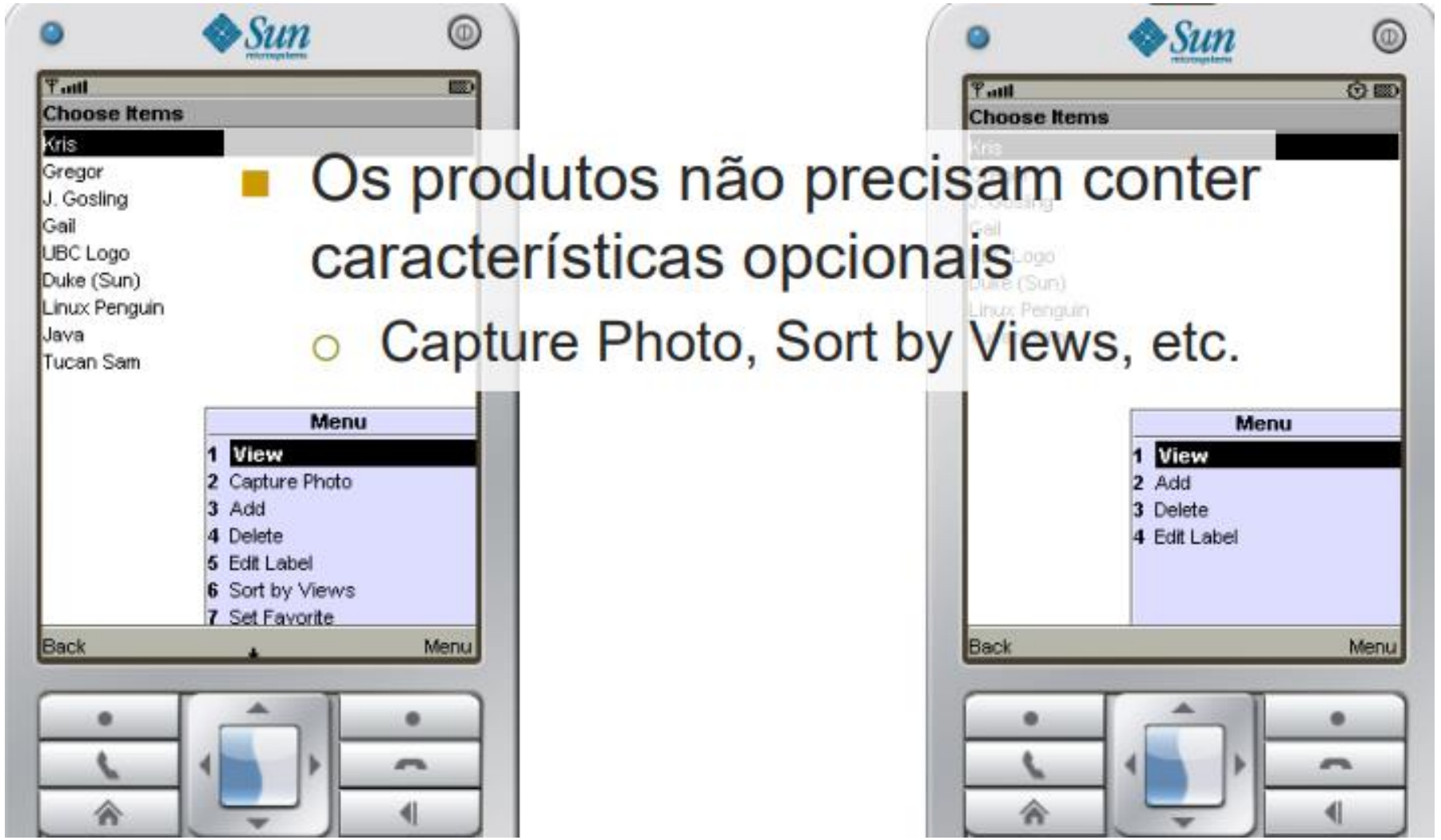



UniCesumar

# Características Mandatórias



# Características Opcionais




- Os produtos não precisam conter características opcionais
  - Capture Photo, Sort by Views, etc.

UniCesumar

# Qualidade de Software

TECNOLOGIA DA INFORMAÇÃO

UniCesumar

# Qualidade de Software

- A qualidade de software tem se aprimorado nos últimos 15 anos
  - Empresas têm adotado novas técnicas
  - Orientação a objetos se difundiu
  - Ferramentas CASE têm sido usadas

- Na manufatura, qualidade significa atender às especificações
  - Em software, a definição não é tão simples

# Adequado à Especificação

- Não é fácil definir qualidade de software como adequado à especificação

- A especificação pode estar ambígua, incompleta ou inconsistente
  - Alguns requisitos podem não aparecer na especificação
  - É difícil especificar todos os requisitos

# Atributos Implícitos de Qualidade

- Alguns atributos são difíceis de serem especificados
  - Mas tem grande efeito na qualidade do sistema

- Exemplos
  - Como garantir segurança dos dados?
  - Como documentar sobre eficiência?
  - Como especificar a facilidade de manutenção?

# Alguns Atributos de Qualidade

| | | |
|---|---|---|
| Segurança | Complexidade | Modularidade |
| Proteção | Robustez | Eficiência |
| Confiabilidade | Adaptabilidade | Portabilidade |
| Facilidade de recuperação | Facilidade de uso | Facilidade de reuso |
| Facilidade de compreensão | Facilidade de testes | Facilidade de aprendizado |

# Equipe de Qualidade

- Idealmente, a equipe de garantia da qualidade deve ser diferente da equipe de desenvolvimento

- O processo de qualidade envolve
  - Definir padrões de processo
  - Monitorar o processo para verificar o adequado uso dos padrões
  - Emitir relatórios para a gerência de projeto e da organização

# Atividades de Gerenciamento

- Garantia de Qualidade
  - Estabelece um framework com os processos e padrões
- Planejamento de Qualidade
  - Seleção dos procedimentos e padrões apropriados ao projeto
- Controle de Qualidade
  - Verifica se os procedimentos e padrões estão sendo seguidos

# Qualidade de Processo

- Acredita-se que a qualidade do processo afeta diretamente a qualidade do produto

- Esta crença veio da indústria de manufatura

  - Em software, a relação entre qualidade de processo e qualidade de produto é complexa

  - Estudos mostram que a relação existe

# Qualidade de Software

- Na manufatura, o processo é altamente automatizado

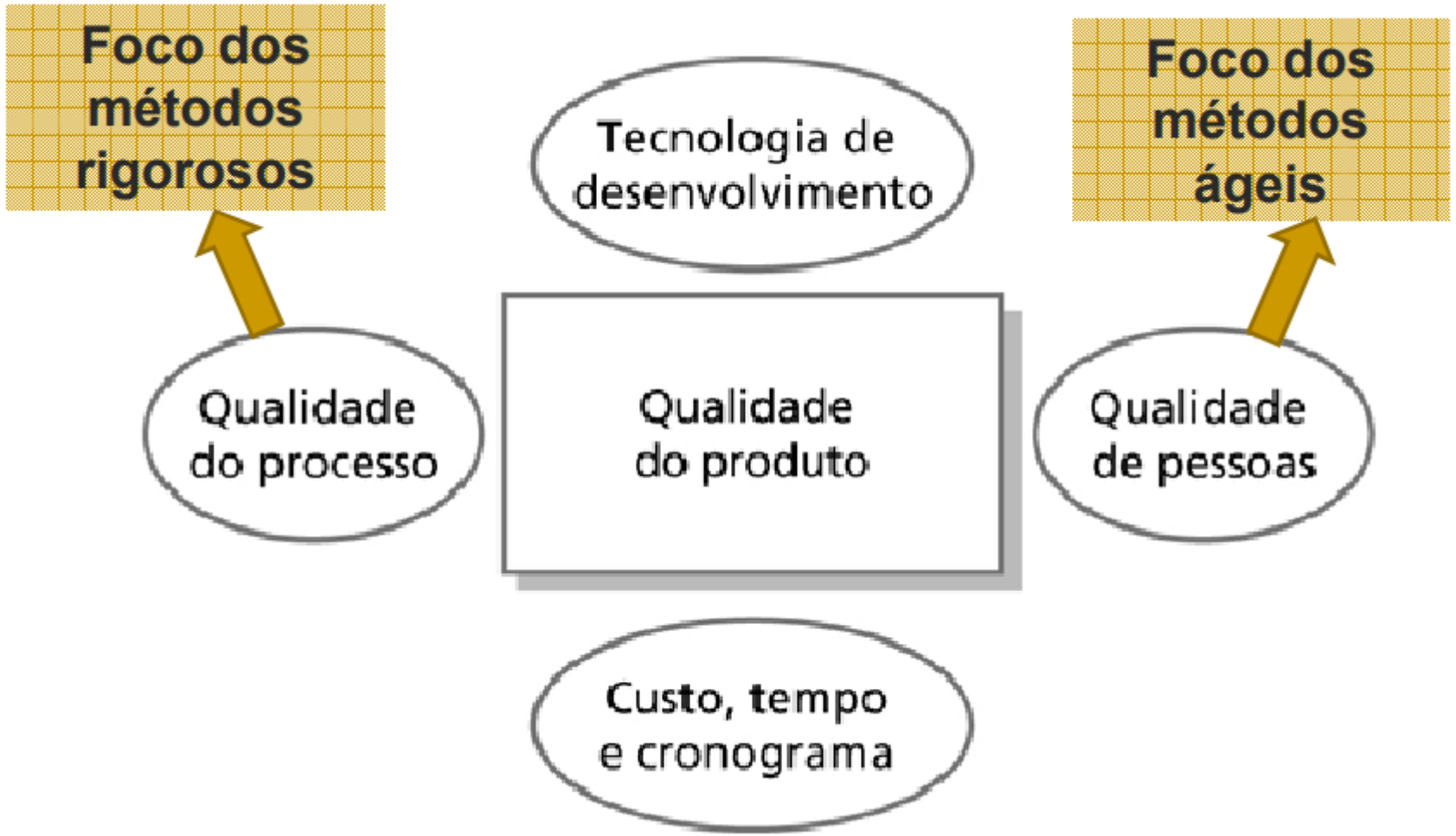
  - Erros de calibração de máquinas causam produtos defeituoso que são facilmente verificados

- Em software, o processo tem grande ingrediente intelectual

  - Erros não são facilmente verificados

  - Qualidade das pessoas é importante

# Qualidade de Software




UniCesumar

# Padrões de Software

- Padrões de produto
  - Aplicam-se ao produto de software que está sendo desenvolvido
  - Padrões de documentação, padrões de codificação, etc.

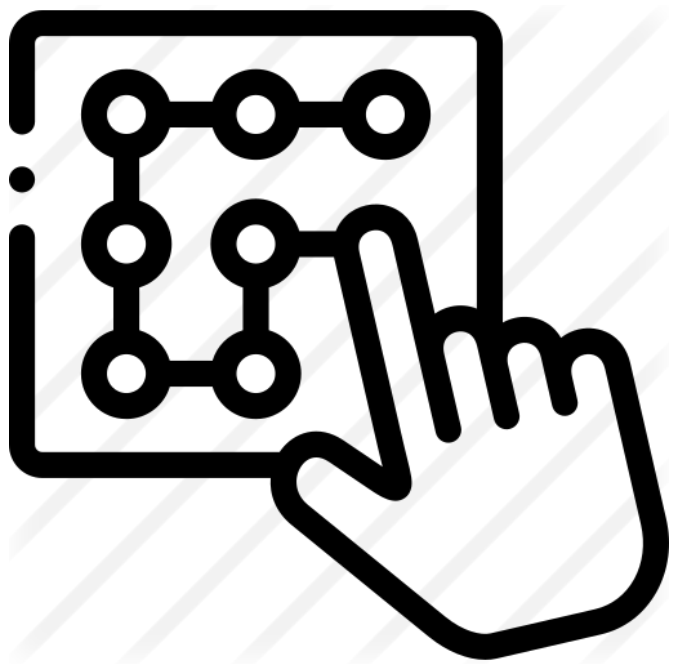
- Padrões de processo
  - Definem as atividades do processo e os seus resultados
  - Processos de validação, ferramentas, etc

# Exemplo: Padrões de Produto

- Formulário de revisão do projeto
- Estrutura do documento de requisitos
- Formato da assinatura de métodos
- Estilo de programação Java
- Formato do plano de projeto
- Formato do formulário de solicitação de mudanças

# Exemplo: Padrões de Processo

- Conduta de revisão de projeto
- Envio de documentos para gerência
- Processo de liberação de versões
- Processo de aprovação do plano de projeto
- Processo de controle de mudanças
- Processo de registro de testes



# Uso de Fato de Padrões

- Alguns cuidados para que os padrões sejam de fato implementados

  - Envolver a equipe de desenvolvimento na escolha dos padrões

  - Revisar os padrões regularmente para refletir mudanças de tecnologia

  - Não incluir apenas o que seguir, mas também o porque de seguir um padrão

  - Prover ferramentas para apoiar a adoção dos padrões

# Engenharia de Software I

UniCesumar