

Qualidade de Software

Eduardo Figueiredo

<http://www.dcc.ufmg.br/~figueiredo>

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
 - Integração de requisitos de diversos *stakeholders*

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?



Adequando à Especificação

- Qualidade do software não implica somente se as funcionalidades foram corretamente implementadas, mas também, dependem de requisitos não funcionais

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 uso adequado dos padrões
 - Emitir relatórios para a gerência de projeto e da organização

Gerência da Qualidade

Gerenciamento de Qualidade

- Garantir com que o nível de qualidade desejado seja alcançado, se preocupando nos diferentes níveis:
 - Organizacional: com a definição de *framework* de processos e padrões que deverão ser adotados
 - Projeto: aplicação dos processos e definição do plano de qualidade

[O Tamanho do Projeto]

- Mesmo em projetos pequenos, o gerenciamento da qualidade é importante
 - Entretanto, ele pode seguir uma abordagem mais informal
- Em sistemas grandes, a gerência da qualidade requer três atividades
 - Garantia de Qualidade
 - Planejamento de Qualidade
 - Controle de Qualidade

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

Garantia da Qualidade (QA)

- A garantia da qualidade de software busca saber
 - Como a qualidade pode ser atingida
 - Se a qualidade foi atingida
- Estabelece os procedimentos e padrões da organização



Planejamento da Qualidade

- É o processo de desenvolvimento de um plano de qualidade para um projeto
- Estabelece os padrões apropriados para um produto e processo
- Documento não deve ser muito longo



Estrutura de Planejamento

1. Descrição do produto, mercado e das expectativas de qualidade
2. Plano com as datas críticas e responsabilidades
3. Descrição dos métodos e serviços usados no desenvolvimento e gerenciamento do produto
4. Definição das metas de qualidade e respectivas justificativas
5. Descrição dos riscos e ações para minimizá-los

Controle de Qualidade

- Envolve o monitoramento do processo de desenvolvimento
- Busca assegurar que os procedimentos e padrões estão sendo de fato aplicados no projeto



Abordagens de Controle

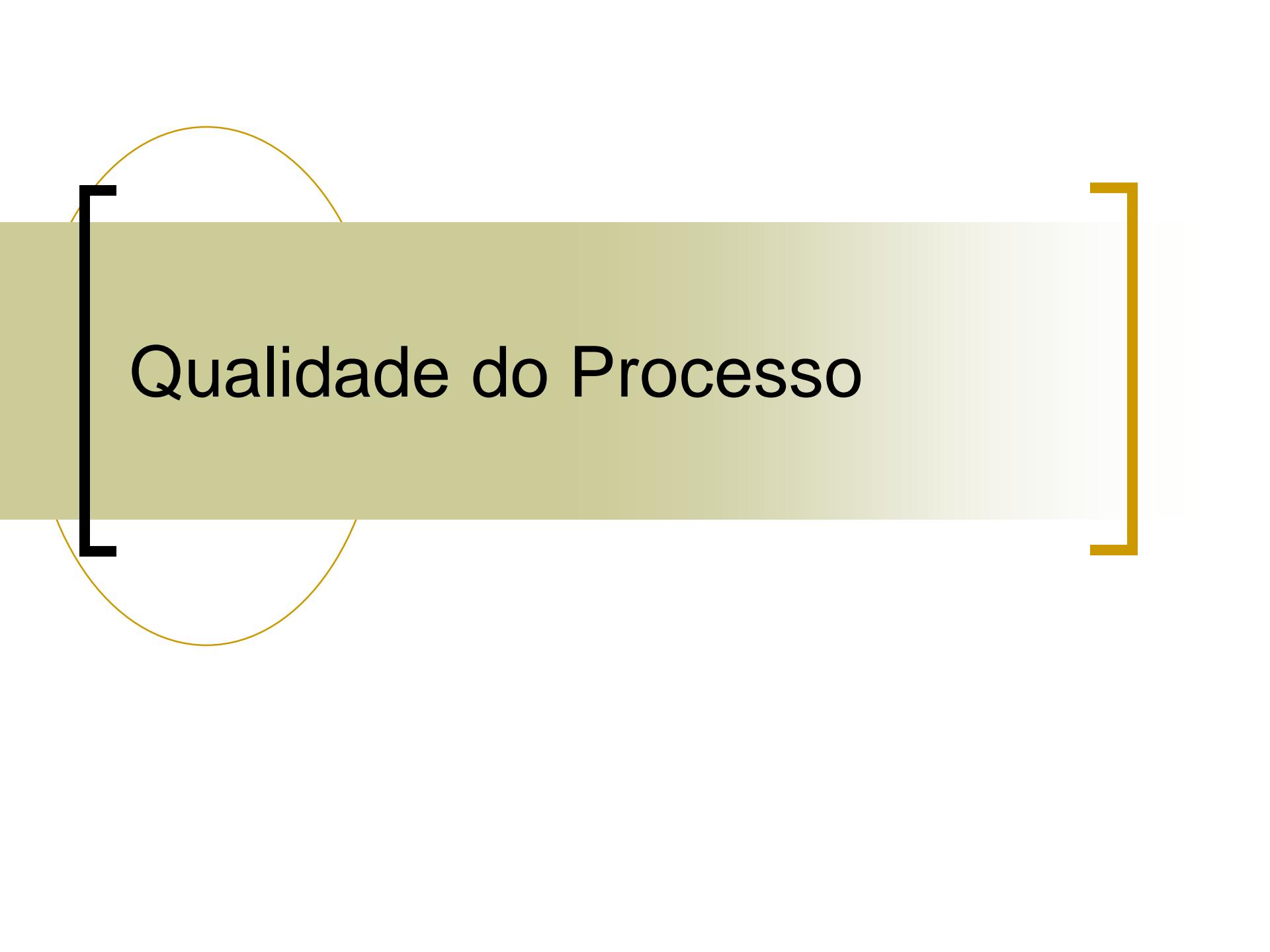
- Avaliação automatizada
 - O produto (ou documentação) é processado automaticamente
 - Métricas são usadas para verificar a qualidade
- Revisão
- Inspeção

Revisão

- Grupo de pessoas examinam o entregáveis a procura de problemas, não conformidades com os padrões e omissões
 - Ex.: código fonte, plano de testes
- Processo público de detecção de erros, sendo necessário criar uma cultura de trabalho que não culpe os indivíduos.

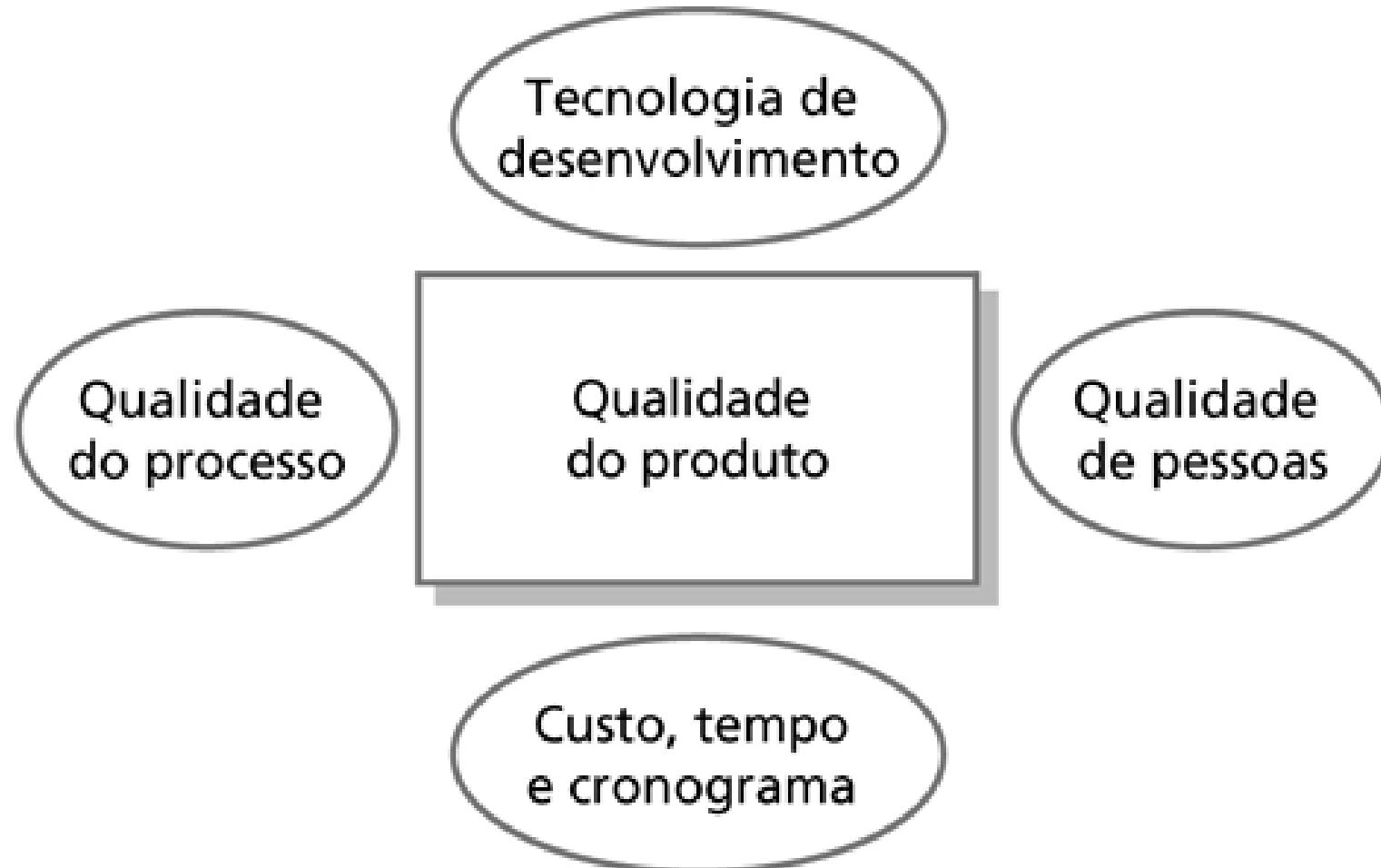
[Inspeção]

- Atividade de revisar o código fonte a procura de defeitos e bugs
- Frequentemente se usam *checklists* de erros comuns de programação, em que cada *checklist* é baseado na linguagem de programação utilizada
- Exemplos: inicialização de variáveis, terminação de *loops*



Qualidade do Processo

Qualidade de Software



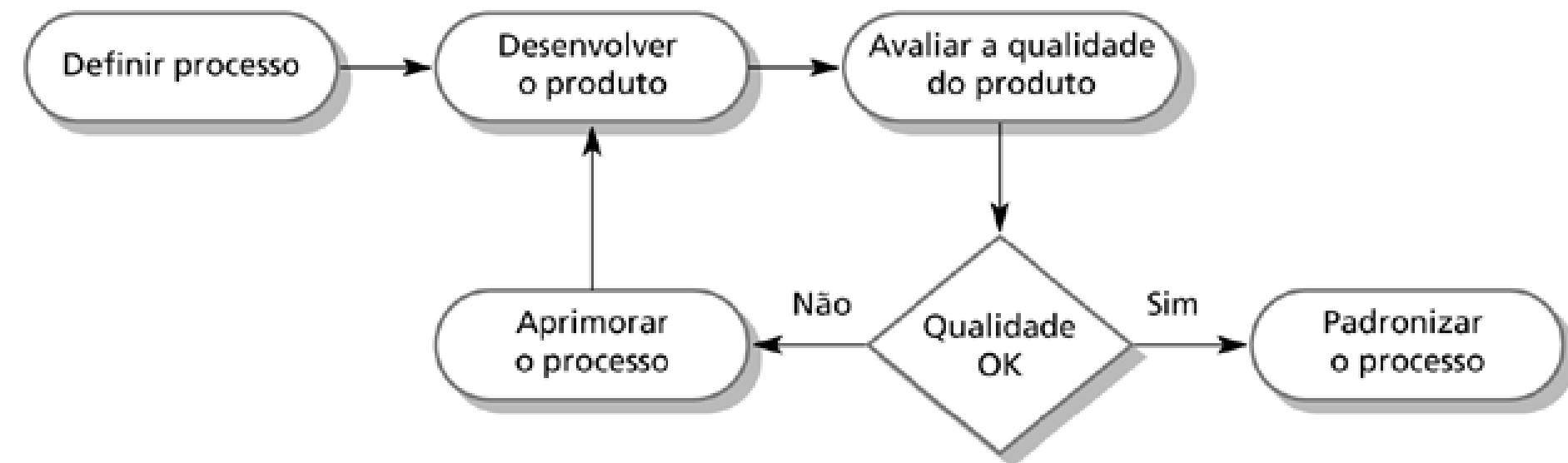
Qualidade de Software

- Na manufatura, o processo é altamente automatizado
 - Erros de calibração de máquinas causam produtos defeituosos que são facilmente verificados
- Em software, o processo tem grande ingrediente intelectual
 - Erros não são facilmente verificados
 - Qualidade das pessoas é importante

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 Baseada no Processo



Grandes e Pequenos Projetos

- Em grandes projetos de software
 - A equipe de desenvolvimento é volátil
 - A qualidade do processo é fator predominante
- Em projetos de pequeno porte
 - Quantidade pequena de pessoas envolvidas
 - A qualidade da equipe é mais importante que a qualidade do processo

Qualidade de Software

**Foco dos
métodos
rigorosos**

Tecnologia de
desenvolvimento

**Foco dos
métodos
ágiles**

Qualidade
do processo

Qualidade
do produto

Qualidade
de pessoas

Custo, tempo
e cronograma

Padrões de Software

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.

Exemplos: 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

Exemplos: 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

[

Importância de Padrões

]

- Documentam o conhecimento das melhores práticas
- Indicam o caminho para se obter qualidade
 - Principalmente aos menos experientes
- Facilitam a comunicação entre os membros da equipe

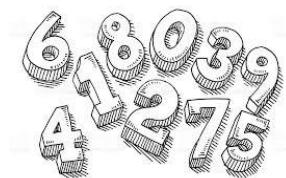
Uso de Fato de Padrões

- 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

Medição de Software

Métricas de Software

- Medições se dedicam a obter um ou mais valores numéricos para um atributo de qualidade
 - Ao comparar os números, é possível tirar conclusões sobre a qualidade do produto
- Uma métrica de software é qualquer medição que se refere ao sistema
 - Medições de tamanho (exemplo, LOC)
 - Número de defeitos relatados, etc.



Tipos de Métricas

- Elas podem ser:
 - Controle: suporta o processo de gerenciamento (exemplo: tempo necessário para reparar os defeitos encontrados)
 - Previsão: ajuda a prever características do software (exemplo: número de operações associadas a um objeto)

[

Por que medir?

]

- Revisão para avaliação da qualidade é uma atividade demorada
 - Geralmente causa atraso na conclusão do projeto
 - Ferramentas devem ser empregadas para acelerar o processo de revisão.
- Métricas podem ser usadas para apoiar a tomada de decisões



[Uso de Medições]

- Medições de software podem ser usadas de duas maneiras
 - Avaliar a qualidade do sistema e fazer previsões gerais sobre ele (exemplo, número de defeitos)
 - Para identificar partes (ou módulos) problemáticas(os)

Adoção pela Indústria

- Muitas empresas ainda não usam medições sistemáticas para avaliar a qualidade
- Algumas razões
 - Os processos das empresas não são maduros o suficiente
 - A ausência de métricas padronizadas
 - Limitado apoio de ferramentas de medição

Problemas com Medições

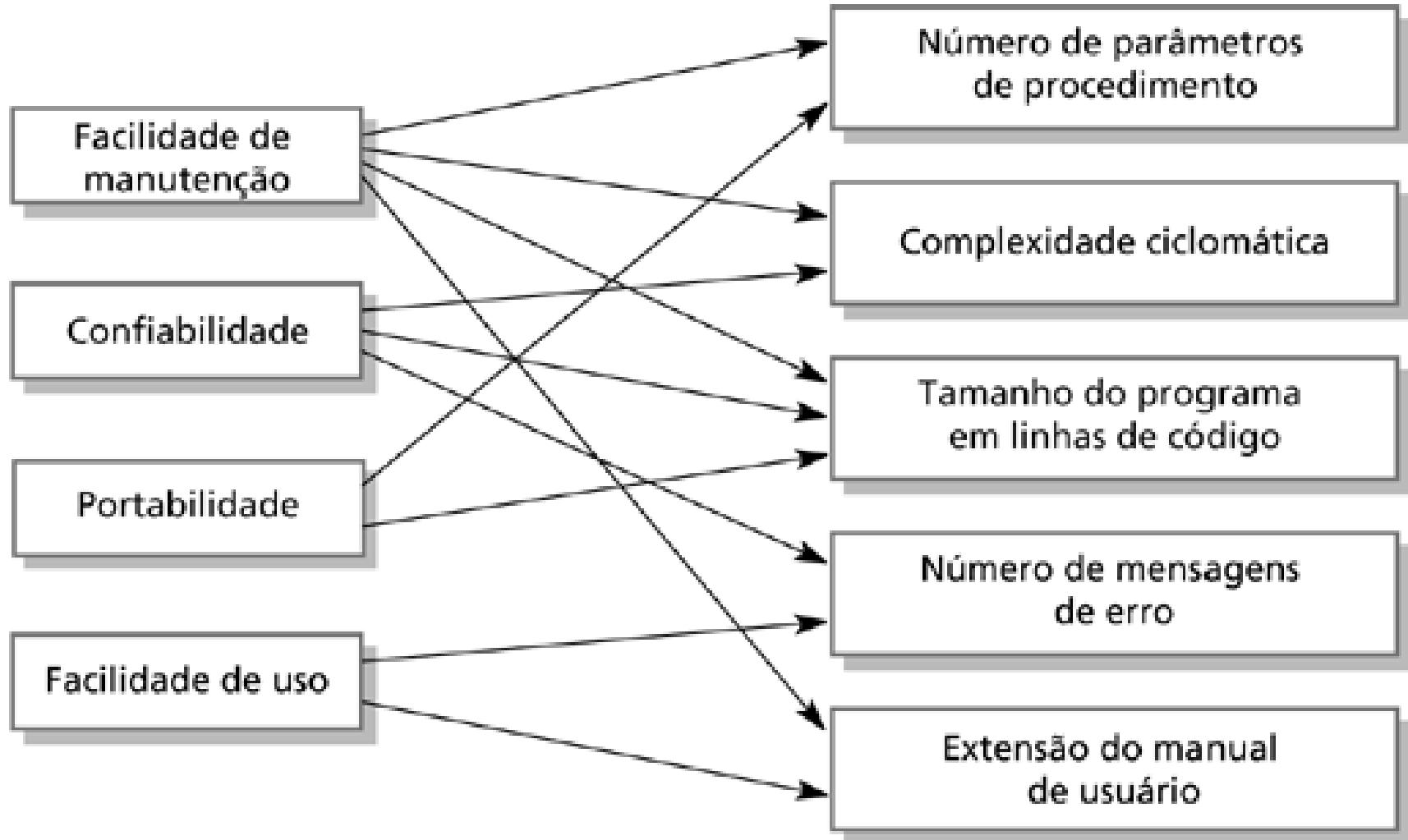
- Geralmente é impossível medir um atributo de qualidade diretamente
 - Atributos de qualidade são fatores externos ao software
 - Métricas medem fatores internos
- Exemplos de atributos de qualidade
 - Facilidade de manutenção
 - Facilidade de uso
 - Confiabilidade

Modelos de Qualidade

Modelos de Qualidade

- Relacionam atributos internos com atributos de qualidade
 - Atributos internos são mais facilmente quantificáveis
- Deveria haver um relacionamento claro e válido entre atributos de qualidade e atributos internos (ideal)

Um Modelo de Qualidade



Validade dos Modelos

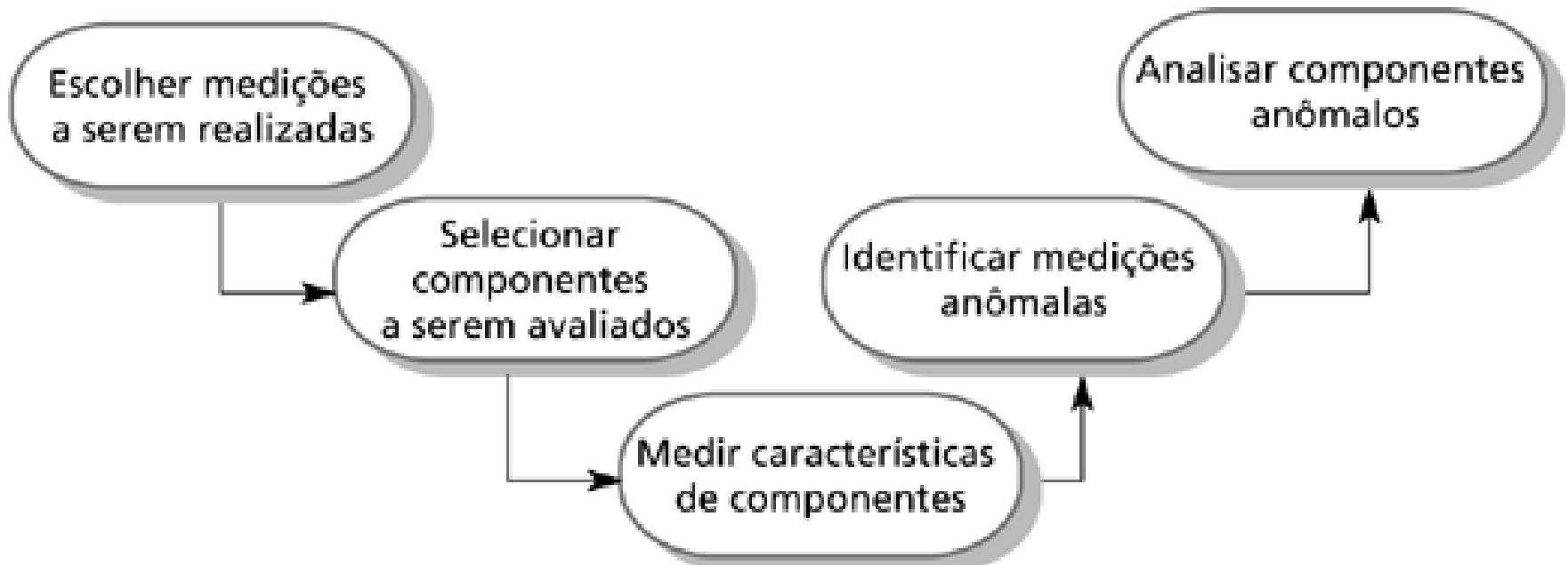
- Três condições devem ser verificadas em modelos de qualidade
 - O atributo interno deve ser precisamente medido
 - Deve haver relacionamentos entre o que podemos medir e o que queremos saber
 - Os relacionamentos são compreendidos e válidos

Processo de Medição

O Processo de Medição

- O processo de medição deve fazer parte do processo de controle da qualidade
 - Utilizam dados históricos de projetos anteriores
- As atividades do processo
 - Escolher medições a serem realizadas
 - Selecionar componentes a serem avaliados
 - Medir características dos componentes
 - Identificar medições anômalas
 - Analisar componentes anômalos

Modelo do Processo



Escolher Medições

- Uma abordagem para escolher as medições é o GQM
 - *Goal-Question-Metric*
- As questões são formuladas para atender um objetivo
- As métricas são escolhidas para responderem as questões

O Modelo de Medição GQM

- Meta (G)
 - Definem o que a organização quer melhorar (exemplo: produtividade)
- Questões (Q)
 - Refinamento dos objetivos em áreas de incertezas (exemplo: linhas de código produzidas podem ser aumentadas?)
- Métricas (M)
 - Medições necessárias para responder as questões (exemplo: LOC por desenvolvedor)

Selecionar Componentes

- Pode não ser necessário (ou desejável) medir todo o sistema
- Estratégias de escolha
 - Escolher um subconjunto representativo de todos os componentes
 - Escolher os componentes particularmente críticos no sistema

Medir os Atributos de Qualidade

- Os componentes selecionados são medidos
- As medidas são associadas aos atributos de qualidade
 - Geralmente envolve uma representação dos componentes
- Ferramentas de medição podem estar incorporadas a outras ferramentas (ou ambientes) de desenvolvimento

Analisar Medições

- Uma vez feita as medições, é preciso compará-las a medições anteriores
 - Dados históricos são utilizados
- A análise deve procurar valores incomuns
 - Ou seja, valores muito altos ou muito baixos para cada métrica

Analizar Componentes

- Se um componente tem valores anômalos, este deve ser examinado
 - A inspeção é responsável por decidir se existe (ou não) problema no componente
- Um valor incomum para um componente não necessariamente significa que o componente tenha baixa qualidade

Análise de Medições

- Nem sempre é óbvio o que os dados significam
 - Entender uma grande quantidade de números é muito difícil
- Estatísticos devem ser consultados, se estiverem disponíveis
- A análise de dados deve levar em conta as circunstâncias locais

Métricas de Produto

Métricas de Produto

- Quantificam atributos internos do software
- Exemplos de atributos
 - Tamanho
 - Acoplamento entre componentes
 - Coesão de um componente, etc.

[

Tipos de Métricas

]

■ Métricas Dinâmicas

- São coletadas por medições realizadas durante a execução do programa (exemplo: tempo de execução)

■ Métricas Estáticas

- São coletadas por medições realizadas na documentação de projeto ou código fonte do programa (exemplo: linhas de código)

Dinâmicas x Estática

- Métricas dinâmicas ajudam a avaliar atributos de qualidade como eficiência e confiabilidade
 - São medidas após o sistema ter sido implementado
- Métricas estáticas ajudam a avaliar atributos como complexidade e facilidade de manutenção
 - Podem ser medidas na fase de projeto

Métricas Estáticas Tradicionais

Algumas Métricas Estáticas

- Fan-in / Fan-out
- Tamanho do código
- Complexidade Ciclomática
- Tamanho do Vocabulário
- Profundidade de Aninhamento

[Fan-in e Fan-out]

■ Fan-in

- Conta o número de funções que chamam uma determinada função
- Valor alto significa grande impacto em mudanças (propagação)

■ Fan-out

- Conta o número de funções chamadas pela função
- Valor alto significa grande complexidade da função

Tamanho e Complexidade

■ Tamanho

- Tamanho tem se mostrado como as métricas mais confiáveis e úteis
- Em geral, quanto maior o componente, mais complexo e propenso a erros ele será

■ Complexidade Ciclomática

- Mede a complexidade de controle do programa (*if*, *while*, *for*, etc.)
- Está relacionada a facilidade de compreensão

Vocabulário e Aninhamento

■ Tamanho do Vocabulário

- Conta a quantidade de identificadores (exemplo, nome de classes) do programa
- Mais identificadores podem significar que eles são mais significativos

■ Profundidade de Aninhamento

- Conta estruturas internas como *if* e *while* aninhados
- Estruturas aninhadas são mais difíceis de se compreender

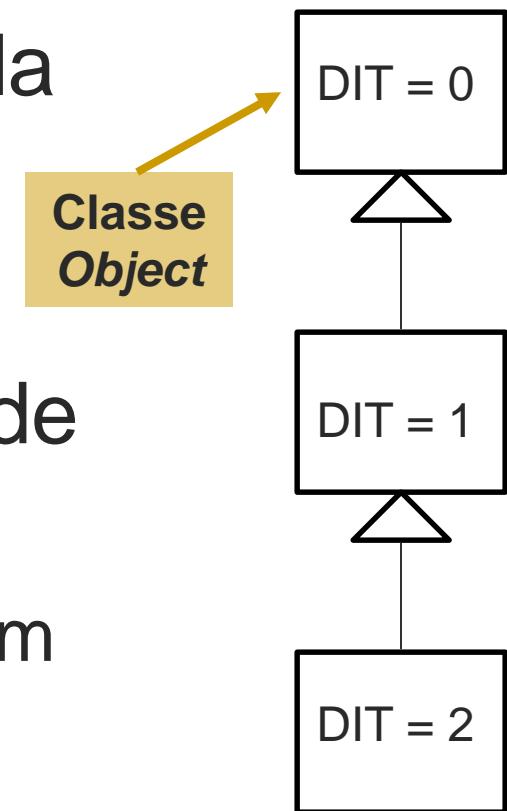
Métricas para Programas Orientados a Objetos

Métricas de Programas OO

- Métricas de Chidamber-Kemerer (CK)
 - Métodos Ponderados por Classes (WMC)
 - Profundidade da Herança (DIT)
 - Número de Filhos (NOC)
 - Acoplamento entre Objetos (CBO)
 - Falta de Coesão em Métodos (LCOM)
- Número de Operações Sobreescritas

Profundidade de Herança (DIT)

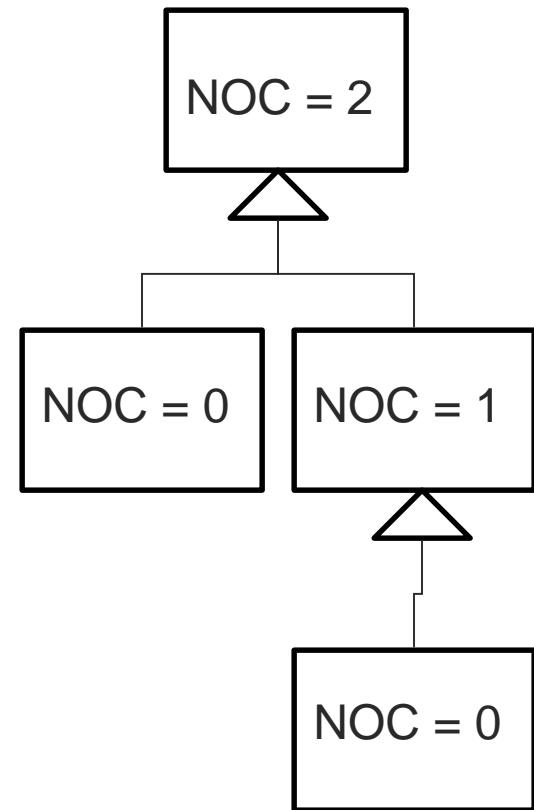
- Representam o número de níveis que uma classe herda métodos e atributos



- Quanto maior a profundidade
 - Mais complexo o projeto
 - Mais difícil de se entender um módulo

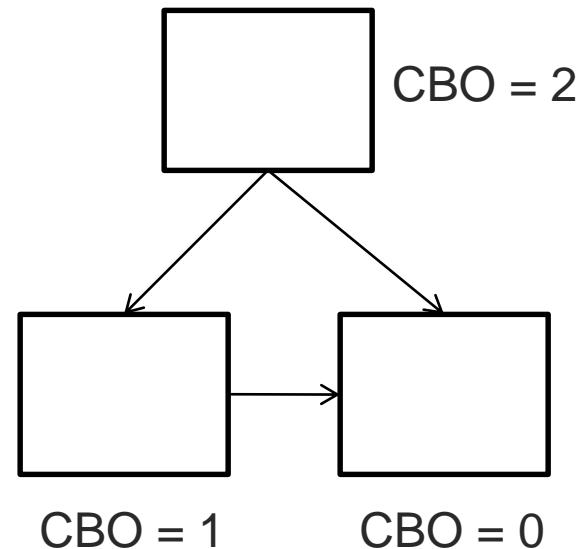
[Número de Filhos (NOC)]

- Conta o número de subclasses diretas
 - Mede a largura da hierarquia de uma classe
- Valor alto, pode indicar maior reuso



Acoplamento entre Objetos (CBO)

- Semelhante a Fan-out
 - Conta classes chamadas por uma classe
- Quanto mais acoplado uma classe
 - Mais difícil de entender e de manter



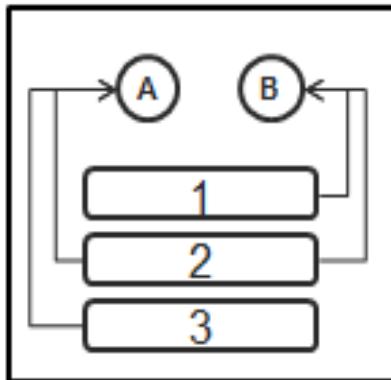
Falta de Coesão (LCOM)

- Mede o quanto os métodos de uma classe acessam atributos em comum
 - Mais atributos em comum, maior coesão, menor perda de coesão (LCOM)
- Diferença entre número de pares de métodos sem atributos compartilhados e número de pares de métodos com atributos compartilhados

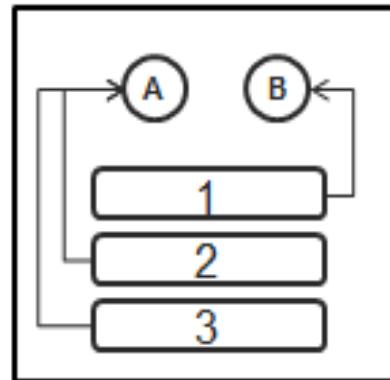
Falta de Coesão (LCOM)

Atributos: A e B

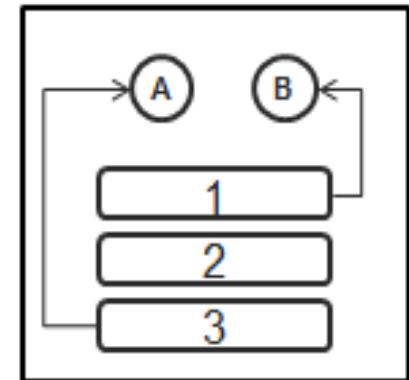
Pares de métodos = $\{(1,2), (1,3), (2,3)\}$



$LCOM = 0 \quad (1 - 2)$



$LCOM = 1 \quad (2 - 1)$



$LCOM = 3 \quad (3 - 0)$

Métricas para Métodos

- Métodos Ponderados por Classes (WMC)
 - Atribui pesos aos métodos de uma classe
 - Uma forma é “pesar” por linhas de código
 - Valores altos indicam complexidade
- Número de Operações Sobrescritas
 - Conta as operações de uma classe que são sobrescritas por subclasses
 - Valores altos indicam problema na hierarquia de herança

[

Bibliografia da Aula

]

- Ian Sommerville. Engenharia de Software, 9^a Edição. Pearson Education, 2011.
 - Cap. 24 Gerenciamento de Qualidade