

Qualidade de Software

Gerenciamento de Qualidade de Software

- Dedicar-se a assegurar que o **nível requerido de qualidade** seja atingido
 - Em um produto de software
- Envolve a definição de padrões e procedimentos apropriados de qualidade e a garantia de que sejam seguidos
- Deve visar o desenvolvimento de uma 'cultura de qualidade'
 - Qualidade vista como uma responsabilidade de todos

O que é Qualidade?

- Qualidade, de maneira simplista, significa que **um produto deve atender às sua especificação**
- Isso é problemático para os sistemas de software
 - Tensão entre os requisitos de qualidade do cliente (eficiência, confiabilidade, etc.) e requisitos de qualidade do desenvolvedor (facilidade de manutenção, reusabilidade, etc.)
 - Alguns requisitos de qualidade são difíceis de especificar de uma maneira não-ambígua
 - Ex. **Facilidade de uso**
 - As especificações de software são, geralmente, incompletas e frequentemente inconsistentes

Escopo do Gerenciamento de Qualidade

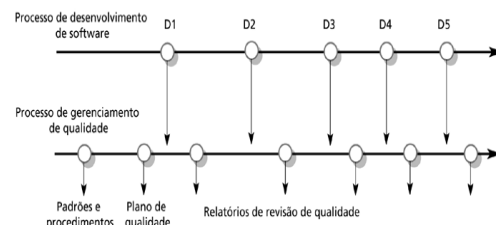
- Gerenciamento de qualidade é particularmente importante para **sistemas grandes e complexos**
- A documentação de qualidade é um registro do progresso
 - Apóia a continuidade do desenvolvimento quando a equipe de desenvolvimento muda
- Para sistemas menores, o gerenciamento de qualidade precisa de menos documentação
 - Foco em uma **cultura da qualidade**

Atividade de Gerenciamento de Qualidade

- **Garantia de qualidade**
 - Estabelece procedimentos e padrões **organizacionais** para qualidade
- **Planejamento de qualidade**
 - Seleciona procedimentos e padrões aplicáveis para um **projeto específico** e o modifica quando necessário.
- **Controle de qualidade**
 - Assegura que os procedimentos e os padrões sejam seguidos pela equipe de desenvolvimento de software.
- O gerenciamento de qualidade deve ser separado do gerenciamento de projeto para assegurar independência
 - Objetivos potencialmente conflitantes!

Gerenciamento de Qualidade e Desenvolvimento de Software

Figura 27.1 Gerenciamento de qualidade e desenvolvimento de software.



Qualidade de Processo e de Produto

- A qualidade de um produto é influenciada pela **qualidade do processo de produção**
Alguns modelos de qualidade partem dessa premissa
- Isso é importante no desenvolvimento de software, visto que os atributos de qualidade de produtos são difíceis de avaliar.
- Contudo, existe uma relação **complexa e pouco compreendida** entre processos de software e qualidade de produto.

Qualidade Baseada em Processo

- Existe uma ligação nítida entre processo e produto nos **bens manufaturados**
- Para o software isso é mais complexo porque:
 - A aplicação de habilidades individuais e a experiência são muito importantes no desenvolvimento de software
 - Fatores externos, como a novidade de uma aplicação ou a necessidade de um cronograma de desenvolvimento acelerado, podem piorar a qualidade do produto.
- Deve-se tomar cuidado para não impor padrões de processo inadequados
 - Esses padrões poderiam reduzir, ao invés de melhorar a qualidade do produto

Padrões de Processo na Prática

- Definir padrões de processo, tais como o modo como as revisões devem ser conduzidas, o gerenciamento da configuração, etc
- Monitorar o processo de desenvolvimento para assegurar que os padrões **estejam sendo seguidos**
- Relatar sobre o processo para a gerência de projeto e para o comprador do software
- Não usar práticas inadequadas simplesmente porque padrões foram estabelecidos

Padrões de Qualidade

- São a chave para o gerenciamento efetivo de qualidade.
- Podem ser internacionais, nacionais, organizacionais ou de projeto.
- Padrões de produto** definem características que todos os componentes devem exibir
 - Ex. um estilo de programação comum.
- Padrões de processo** definem características relativas aos processos de software

Importância dos Padrões

- Englobam as **melhores práticas**
 - Evitam a repetição de erros do passado.
- São um arcabouço para os processos de garantia de qualidade
- Padrões fornecem continuidade
 - Novos funcionários podem compreender a organização pela compreensão dos padrões que são adotados

Exemplos de Padrões de Produto e de Processo

Tabela 27.1 Padrões de produto e de processo

Padrões de produto	Padrões de processo
Formulário de revisão de projeto	Condução de revisão de projeto
Estrutura de documento de requisitos	Envio do documento para o CM
Formato do cabeçalho de método	Processo de liberação de versão
Estilo de programação em Java	Processo de aprovação de plano de projeto
Formato de plano de projeto	Processo de controle de mudança
Formulário de solicitação de mudança	Processo de registro de teste

Problemas com Padrões

- Podem **não ser vistos como relevantes** ou atualizados pelos engenheiros de software
- Envolvem, freqüentemente, **muita burocracia**
- Difícil manter a documentação associada
 - **Ferramentas** são fundamentais!
 - Normalmente, a documentação termina ficando **defasada**

Desenvolvimento de padrões

- Envolver os engenheiros na elaboração desenvolvimento
 - Eles devem compreender as **razões** de um padrão
- Revisar padrões e seu uso regularmente
 - Padrões podem se tornar rapidamente desatualizados e isso reduz sua credibilidade
 - Incorporar **novas** "melhores práticas"
- Padrões devem ter ferramentas de apoio
 - Trabalho padronizado em excesso é a mais importante reclamação contra os padrões

Medições e Métricas de Software

- A medição de software se dedica a derivar um **valor numérico** para algum atributo de um produto ou de processo de software
- Permite comparações **objetivas** entre técnicas e processos
- Embora algumas empresas tenham introduzido programas de medição, a maioria das organizações ainda não as usam de forma sistemática
- Existem poucos padrões estabelecidos nessa área

Métricas de Software

- Contagem de determinados elementos de um sistema de software, processo ou documento
 - Linhas de código em um programa, referências a uma variável global
- Permitem que o software e o processo de software sejam **quantificados**
- Podem ser usadas para prever atributos de produto e para controlar o processo de software.
- As métricas de produto podem ser usadas para **previsões gerais** ou para **identificar componentes anômalos**

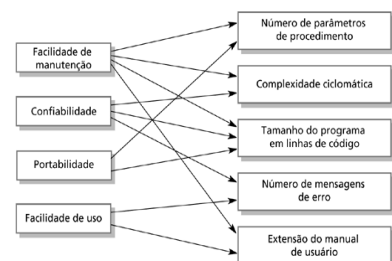
Suposições de Métricas

- Uma propriedade do software **pode ser medida**
- Existe um **relacionamento** entre o que podemos medir e o que queremos conhecer
 - Podemos somente medir atributos **internos**, mas estamos, muitas vezes, mais interessados em atributos **externos** de software
 - Ex. Acoplamento menor => maior manutenibilidade?
- Esse relacionamento foi **formalizado e validado**.
 - Pode ser difícil, a partir do que pode ser medido, inferir as qualidades desejáveis do sistema

Atributos Internos e Externos

Figura 27.6

Relacionamentos entre atributos de software internos e externos.



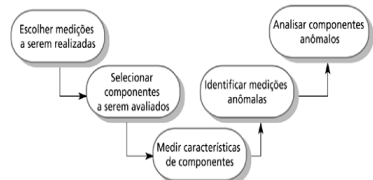
O Processo de Medição

- Um processo de medição de software pode ser parte do controle de qualidade
 - Algumas métricas são difíceis de se coletar de forma automatizada
 - Ex. Separação de interesses
- Os dados coletados durante este processo devem ser **mantidos** como um recurso da organização.
 - Torna possíveis **comparações entre projetos**

Processo de Medição de Produto

Figura 27.7

Processo de medição de produto.



Recomendações para o Processo de Medição

- Não colete dados desnecessários
 - As questões a ser respondidas devem ser decididas previamente e os dados necessários identificados
- Conte às pessoas porque os dados estão sendo coletados
 - Não deve ser parte da avaliação de pessoal
- Não dependa da memória
 - Colete dados quando são gerados, e não depois que um projeto foi finalizado

Métricas de Produto

- Uma métrica deve ser um **previsor de qualidade de produto**.
- Classes de métrica de produto
 - Dinâmicas**, coletadas em tempo de execução
 - Ajudam a avaliar a eficiência e a confiabilidade
 - Estáticas**, coletadas em tempo de compilação
 - Ajudam a avaliar a confiabilidade, a facilidade de compreensão e a facilidade de manutenção.
- Algumas métricas são coletadas a partir de informações **sobre** o produto
 - Ex. Número de **bugs conhecidos**

Métricas Estáticas e Dinâmicas

- Métricas dinâmicas são **diretamente relacionadas** aos atributos de qualidade de software
 - É relativamente fácil medir o tempo de resposta de um sistema (atributo de desempenho) ou a frequência com que ele falha (atributo de confiabilidade).
- Métricas estáticas têm um **relacionamento indireto** com atributos de qualidade
 - É necessário estabelecer um relacionamento entre essas métricas e as propriedades, tais como complexidade, facilidade de compreensão e de manutenção.

Métricas de Produto de Software

Tabela 27.4 Métricas estáticas do produto de software

Métrica de software	Descrição
Fan-in/Fan-out	Fan-in é uma medida do número de funções ou métodos que chamam alguma outra função ou método (digamos X). Fan-out é o número de funções chamadas pela função X. Um valor alto para fan-in significa que X está firmemente acoplado com o resto do projeto, e mudanças em X terão grande impacto. Um alto valor para fan-out sugere que a complexidade geral de X pode ser alta devido à complexidade da lógica de controle necessária para coordenar os componentes chamados.
Extensão de código	É uma medida do tamanho de um programa. Geralmente, quanto maior for o tamanho do código de um componente, mais complexo e propenso a erros esse componente será. A extensão de código foi mostrada como uma das métricas mais confiáveis de previsão de propensão a erros em componentes.
Complexidade ciclomática	É uma medida da complexidade de controle de um programa. Essa complexidade de controle pode estar relacionada à facilidade de compreensão do programa. Explico como calcular a complexidade ciclomática no Capítulo 22.
Extensão de identificadores	É uma medida da extensão média de identificadores distintos em um programa. Quanto maiores forem os identificadores, mais eles serão significativos e, por isso, mais compreensíveis será o programa.
Profundidade de aninhamento de declarações condicionais	É uma medida da profundidade de aninhamento de declarações IF em um programa. As declarações IF profundamente aninhadas são difíceis de compreender e são potencialmente propensas a erros.
Índice de fog	É uma medida da extensão média das palavras e das sentenças em documentos. Quanto maior for o valor para o índice de fog, mais difícil será a compreensão de documentos.

Métricas Orientadas a Objetos

Tabela 27.5 Métricas orientadas a objetos

Métrica orientada a objetos	Descrição
Profundidade de árvore de herança	Representa o número de níveis discretos na árvore de herança da qual as subclasses herdam atributos e operações (métodos) das superclasses. Quanto maior for a profundidade da árvore de herança, mais complexo será o projeto. Muitas classes de objeto podem ser compreendidas ao se compreender as classes de objeto das folhas da árvore.
Fan-in/fan-out de método	Está diretamente relacionado com fan-in e fan-out, conforme descrito na Tabela 27.4, e significa essencialmente a mesma coisa. Contudo, pode ser apropriado fazer uma distinção entre chamadas de outros métodos dentro do objeto e chamadas com base em métodos externos.
Métodos ponderados por classe	Este é o número de métodos incluídos em uma classe, ponderados pela complexidade de cada método. Portanto, um método simples pode ter uma complexidade de 1 e um método grande e complexo pode ter um valor muito mais alto. Quanto maior for o valor dessa métrica, mais complexa será a classe de objeto. Os objetos complexos são, provavelmente, mais difíceis de compreender. Eles podem não ser logicamente coerentes, de modo que não podem ser reusados eficazmente como superclasses em uma árvore de herança.
Número de operações sobrescritas	Este é o número de operações, em uma superclasse, sobrescritas em uma subclasse. Um valor alto dessa métrica indica que a superclasse usada pode não ser uma classe-mãe apropriada para a subclasse.

P

© 2006

© 2006 Pearson Education, Inc.

Engenharia de Software, 8ª edição, Capítulo 27

Slide 27

Análise de Medições

- Nem sempre é óbvio o que os dados significam
 - A **análise** de dados coletados é **muito difícil**
- Estatísticos profissionais devem ser consultados, se estiverem disponíveis
- É muito importante usar métricas **experimentalmente validadas**

PEARSON
Prentice
Hall

© Ian Sommerville 2006

Engenharia de Software, 8ª edição, Capítulo 27

Slide 26