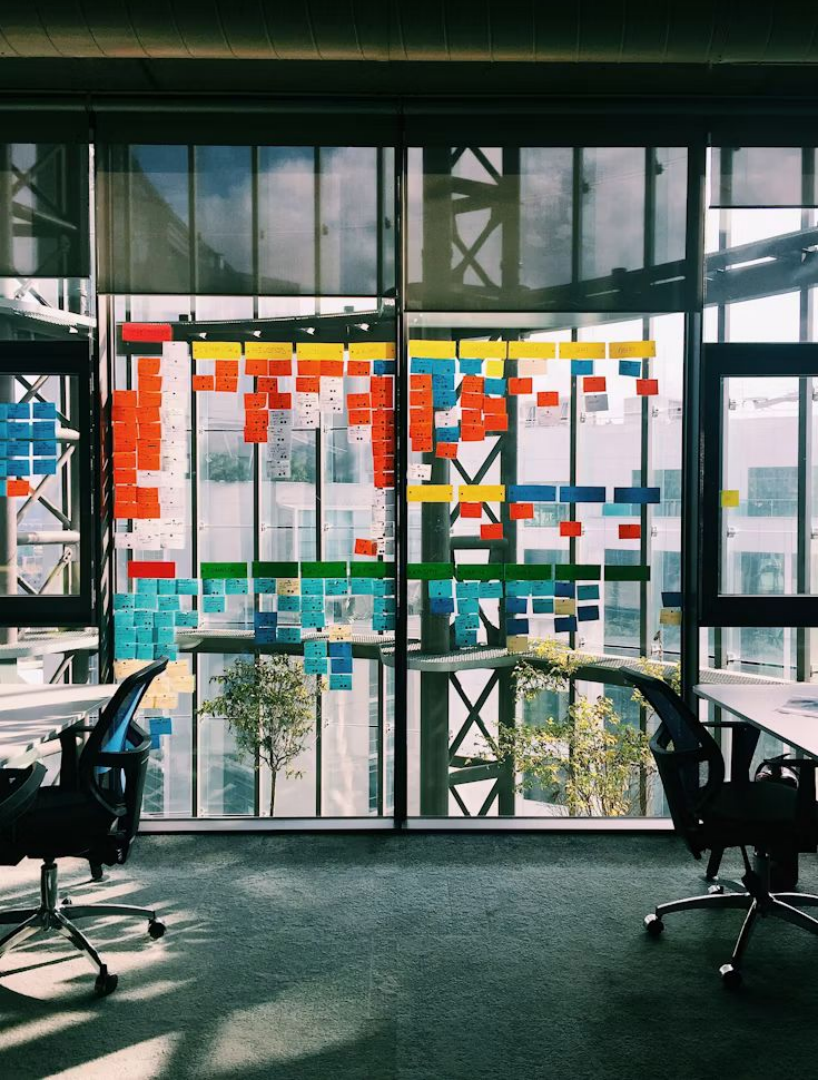


5954014 - Engenharia de Software

Aula 06 - Métricas e Indicadores

Prof. Dr. Denis Martins

DCM | FFCLRP | USP

Objetivos de Aprendizagem

01

Diferenciar medida, métrica e indicador

Compreender as definições fundamentais e as distinções entre estes três conceitos essenciais.

02

Identificar métricas de produto, processo e projeto

Classificar as diferentes métricas de acordo com o seu foco e aplicação no ciclo de desenvolvimento.

03

Diferenciar medidas diretas e indiretas

Classificar as medições de acordo com a forma de obtenção, identificando atributos internos e externos do software.

04

Interpretar métricas dentro de um contexto

Analisar os resultados obtidos considerando as particularidades e o ambiente do projeto.

05

Reconhecer limitações do uso isolado de métricas

Compreender os riscos de decisões baseadas em dados quantitativos sem análise qualitativa complementar.

Motivação

O projeto está indo bem?

01

Mais entregas
Mais funcionalidades foram de fato entregues nas últimas 4 semanas.

02

Mais defeitos
O volume de bugs e problemas encontrados também aumentou.

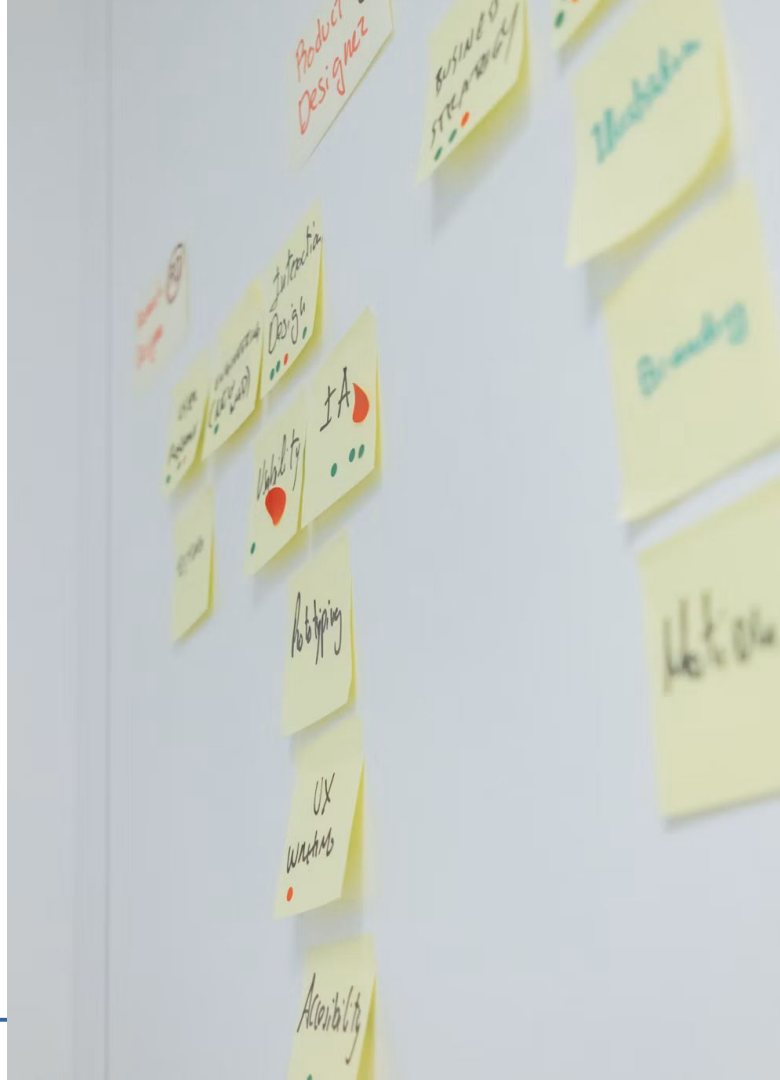
03

Tempo maior
O tempo necessário para concluir novas funcionalidades aumentou.

04

Mais esforço
A equipe passou a trabalhar mais horas para cumprir as entregas.

Afirmação do Gerente
"Nossa produtividade melhorou porque estamos entregando mais funcionalidades."



A conclusão do gerente é válida?

O que você **precisaria** saber antes de responder?

Falta contexto

Qual é o tamanho do sistema? Quantos defeitos havia anteriormente?
Em qual etapa foram encontrados?
Qual é a gravidade? Quanto esforço foi utilizado?

Dados Não Falam Sozinhos

37

defeitos encontrados nesta versão

Bom? Ruim?

Muito? Pouco?

Melhor ou pior que antes?

Precisamos de contexto.

Uma métrica isolada é apenas um número sem significado real.

Para tomar decisões corretas na Engenharia de Software, devemos cruzar dados históricos, complexidade do produto e o esforço empregado pela equipe.

Medir Software: Para Quê?

Qualidade

Indicar a Qualidade

Obter parâmetros claros sobre a qualidade final do produto entregue.

Produtividade

Avaliar Produtividade

Analisar a eficiência e o rendimento das equipes de desenvolvimento.

Processo

Avaliar o Processo

Identificar melhorias contínuas no fluxo de trabalho e ciclo de vida.

Inovação

Novos Métodos

Avaliar os benefícios práticos de novas ferramentas e metodologias.

Gestão

Monitoração e Controle

Apoiar o gerenciamento ativo do projeto com base em dados concretos.

Futuro

Futuras Estimativas

Fornecer dados históricos para planejar prazos e custos mais precisos.

Propósito da Medição: **Medimos para compreender e decidir.**

O que devemos medir?

Como devemos interpretar as medidas obtidas?

Medida, Métrica e Indicador

Esses três conceitos estão relacionados, mas **não são sinônimos**.

Medida

Um **valor** observado diretamente, sem tratamento ou contextualização prévia.

Métrica

Uma medida quantitativa ou uma **relação** calculada entre diferentes medidas.

Indicador

Uma métrica interpretada dentro de um **contexto** específico para apoiar tomadas de decisão.

Medida

O que é?

Uma **medida** representa uma quantificação direta de algum atributo de software, processo ou produto.

Pergunta Respondida

A medida responde essencialmente à pergunta:

Quanto?

Exemplos:

37 defeitos encontrados

120 horas trabalhadas

18 funcionalidades implementadas

240 testes executados

34 arquivos modificados

Foco do Conceito:

Dado Bruto

Métrica

Uma **métrica** relaciona uma ou mais medidas para representar uma característica de interesse de forma quantitativa.

Exemplo 1: Qualidade de Testes

Foram executados: **200 testes**
Obtiveram sucesso: **180 aprovados**

Fórmula:

Taxa = (180 / 200) × 100

Resultado: **90% de aprovação**

Exemplo 2: Densidade de Defeitos

Uma versão possui: **20 funcionalidades**
Foram encontrados: **40 defeitos**

Fórmula:

Densidade = 40 / 20

Resultado: **2 defeitos / func.**

Mas... Esse número sozinho já permite tomar uma decisão?

Lembre-se: métricas isoladas mostram o comportamento, mas ainda carecem do contexto de negócios para virar indicadores.

Modelo SMART para Definição de Métricas

Uma boa métrica deve ser definida de forma que seja **útil, compreensível e acionável**.

Critério	Significado	Exemplo
S	**Specific (Específica)** Deixa claro **o que** será medido.	*Defeitos críticos encontrados após a entrega*
M	**Measurable (Mensurável)** Pode ser **quantificada** em números.	*Número de defeitos por versão*
A	**Achievable (Atingível)** A meta associada deve ser **realista**.	*Reduzir defeitos críticos de 12 para 8*
R	**Relevant (Relevante)** Deve apoiar um **objetivo** do projeto.	*Melhorar a qualidade das entregas*
T	**Time-bound (Temporal)** Deve possuir um **período** definido.	*Alcançar a meta até o final do semestre*

Modelo SMART para Definição de Métricas

Uma boa métrica deve ser definida de forma que seja **útil, compreensível e acionável**.

Critério	Significado	Exemplo
S	**Specific (Específica)** Deixa claro **o que** será medido.	*Defeitos críticos encontrados após a entrega*
M	**Measurable (Mensurável)** Pode ser **quantificada** em números.	*Número de defeitos por versão*
A	**Achievable (Atingível)** A meta associada deve ser **realista**.	*Reduzir defeitos críticos de 12 para 8*
R	**Relevant (Relevante)** Deve apoiar um **objetivo** do projeto.	*Melhorar a qualidade das entregas*
T	**Time-bound (Temporal)** Deve possuir um **período** definido.	*Alcançar a meta até o final do semestre*

Exemplo Prático SMART

"Reduzir em 30% o número de defeitos críticos encontrados após a implantação até o final das próximas 3 versões."

Por que importa? O SMART evita métricas vagas como "*melhorar a qualidade*", transformando objetivos em critérios que podem ser acompanhados e avaliados.

Em Engenharia de Software, as métricas devem apoiar a compreensão, avaliação e decisão sobre:

- **Produto**
- **Processo**
- **Projeto**

Cuidado com Métricas Isoladas

Considere as duas equipes a seguir:

Equipe A
20 func. entregues
40 defeitos

Equipe B
14 func. entregues
8 defeitos

Qual equipe teve melhor desempenho?

Cuidado com Métricas Isoladas

Considere as duas equipes a seguir:

Equipe A
20 func. entregues
40 defeitos

Equipe B
14 func. entregues
8 defeitos

Depende...

- Complexidade e criticidade das entregas
- Tamanho da equipe e esforço consumido
- Prazo e maturidade do produto

Qual equipe teve melhor desempenho?

Uma métrica isolada raramente conta toda a história.

Métricas Precisam de Contexto

Um número isolado ganha significado real e relevância estratégica quando confrontado com:

- **Um objetivo** definido no planejamento
- **Uma versão** ou **período anterior**
- **Uma linha de base** ou dados históricos
- **Um comportamento esperado** do sistema

Exemplo Comparativo

A diferença no impacto da informação:

Dado Isolado: 12 defeitos
Dado Contextualizado: 12 defeitos, redução em comparação com a versão anterior

Versão	Aprovação
V1	**97%**
V2	**94%**
V3	**89%**
V4	**82%**

Métricas Podem Gerar Comportamentos Indesejados

Cenário Hipotético

"A produtividade dos programadores será avaliada pela quantidade de código produzido."

O que pode acontecer?

- **Código desnecessariamente grande:** Estímulo à verbosidade em vez de soluções elegantes.
- **Menor incentivo à simplificação:** Projetos mais complexos e difíceis de manter no longo prazo.
- **Refatoração vista como improdutiva:** Reduzir linhas de código passa a ser punido financeiramente ou na avaliação.
- **Qualidade sacrificada:** Foco em volume gera débitos técnicos gritantes no produto.

Métricas Podem Gerar Comportamentos Indesejados

Cenário Hipotético

"A produtividade dos programadores será avaliada pela quantidade de código produzido."

O que pode acontecer?

- **Código desnecessariamente grande:** Estímulo à verbosidade em vez de soluções elegantes.
- **Menor incentivo à simplificação:** Projetos mais complexos e difíceis de manter no longo prazo.
- **Refatoração vista como improdutiva:** Reduzir linhas de código passa a ser punido financeiramente ou na avaliação.
- **Qualidade sacrificada:** Foco em volume gera débitos técnicos gritantes no produto.

O comportamento induzido pela métrica também deve ser considerado.

Indicador

Um **indicador** é uma métrica ou combinação de métricas utilizada para compreender situações, **identificar padrões** e **viabilizar intervenções** estratégicas.

Utilizado para:

- Compreender uma situação
- Observar uma tendência
- Identificar um problema
- Apoiar uma decisão

Exemplo

A densidade de defeitos aumentou nas últimas versões de:

1,2 para 2,0 defeitos/KLOC

Possível Decisão

Investigar o processo de desenvolvimento antes da próxima entrega.

Medidas Diretas e Indiretas

Medidas Diretas

O atributo de interesse é observado ou mensurado de forma direta, sem necessidade de outras variáveis.

Exemplos na Literatura:

- Custo
- Esforço (pessoa-horas)
- Linhas de código (LOC)
- Velocidade de execução
- Memória utilizada
- Número de erros ou defeitos

Exemplo Prático:
*"O projeto consumiu **620 pessoa-horas**."*

Medidas Indiretas

O atributo não é observado diretamente, sendo inferido por meio de cálculos ou combinação de outras medidas.

Exemplos na Literatura:

- Funcionalidade
- Qualidade
- Complexidade
- Eficiência
- Confiabilidade
- Manutenibilidade

Por que são indiretas?
Normalmente precisamos combinar dados brutos e interpretar diferentes métricas para avaliar o atributo real.

Métricas: Processo × Projeto × Produto

Processo

Foco na engenharia de software e fluxo de trabalho

Pergunta-chave:

Quanto retrabalho ocorreu durante o desenvolvimento?

Projeto

Foco no planejamento, prazos e recursos

Pergunta-chave:

Quanto esforço já foi consumido pelas equipes?

Produto

Foco na qualidade e características do entregável

Pergunta-chave:

Quantos defeitos ainda existem na versão atual?

Visão Integrada

As três perspectivas **se complementam** para permitir diagnósticos precisos e decisões táticas inteligentes durante todo o ciclo de vida do software.

Métricas de Processo

Foco na Efetividade

Fornecem informações valiosas sobre a **efetividade do processo de Engenharia de Software**.

O processo é medido buscando:

- **Compreendê-lo:** Estabelecer uma linha de base clara para o fluxo de trabalho.
- **Avaliar seu desempenho:** Analisar a qualidade e consistência das entregas.
- **Identificar melhorias:** Descobrir gargalos e pontos de otimização contínua.

Exemplos: Esforço & Custo (mensuração de horas dedicadas e investimento financeiro direto no ciclo de vida do software), **Correção & Retrabalho** (acompanhamento do tempo para sanar falhas e o impacto das refatorações não planejadas).

Métricas de Projeto

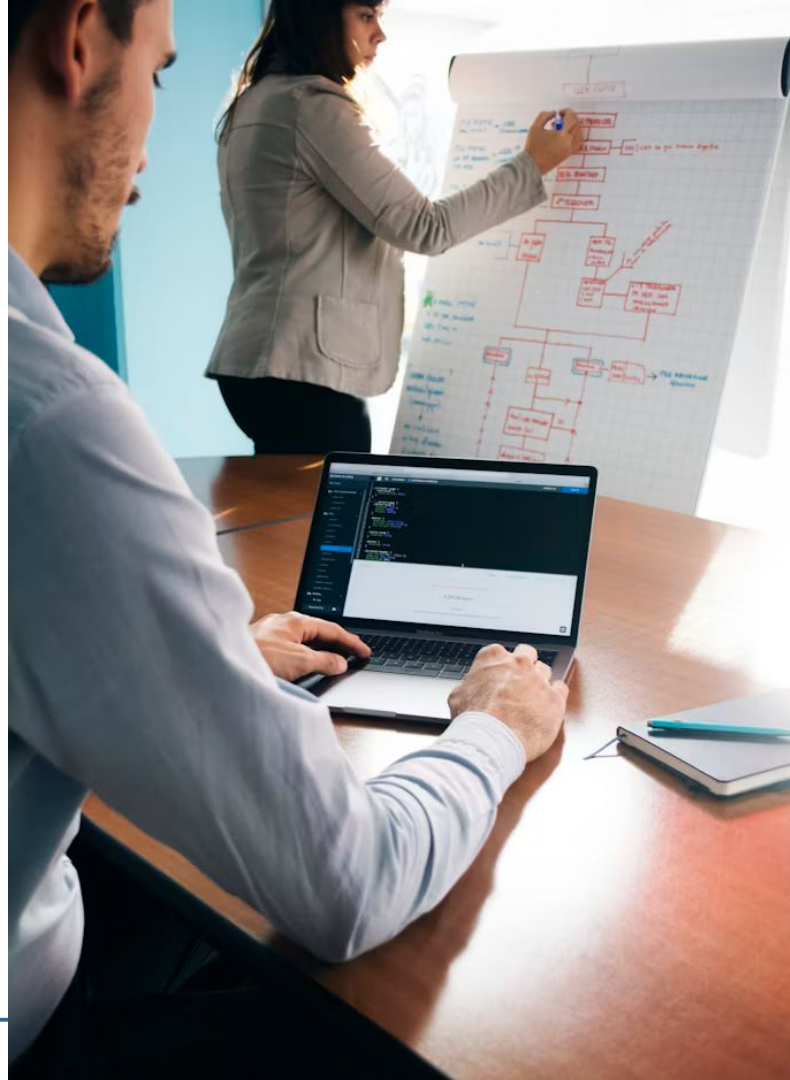
São utilizadas para acompanhar e adaptar o **trabalho de um projeto específico**.

Auxiliam na observação de:

- **Esforço consumido:** Tempo dedicado pelas equipes de desenvolvimento.
- **Trabalho realizado:** Funcionalidades e tarefas concluídas.
- **Recursos utilizados:** Alocação de insumos e orçamento planejado.
- **Progresso e problemas:** Andamento das atividades e impedimentos.

Objetivo Principal

Apoiar decisões táticas durante a execução do projeto.



Métricas de Produto

Relacionam-se às características do **software produzido** durante o projeto.

Exemplos Diretos

Atributos observados diretamente no código ou execução:

- **Linhas de código** (LOC)
- **Velocidade** de execução
- **Memória** utilizada
- **Número** de defeitos

Características Indiretas

Atributos inferidos pela combinação de dados:

- **Complexidade**
- **Eficiência**
- **Confiabilidade**
- **Manutenibilidade**
- **Qualidade**

Exemplo

projeto	esforço	$	KLOC	pags.docum.	defeito	pessoa-mês
projA-01	24	168	12.1	365	29	3
projB-04	60	440	27.2	1224	86	5
projC-03	48	314	20.2	1050	64	6

PRODUTIVIDADE = KLOC / pessoas-mês

QUALIDADE = erros (defeito) / KLOC

CUSTO = $ / LOC

DOCUMENTAÇÃO = pags.docum. / KLOC

Qual equipe foi mais efetiva?

Métricas Absolutas (Cenário Inicial)

À primeira vista, uma equipe parece ter encontrado muito mais problemas:

- **Equipe A:** **342 defeitos**
- **Equipe B:** **184 defeitos**

Ainda não sabemos!

Os projetos possuem tamanhos e complexidades muito diferentes:

Projeto A: 342 defeitos em **150 KLOC**

Projeto B: 184 defeitos em **50 KLOC**

Precisamos de uma métrica normalizada para comparar de forma justa.

Qual equipe foi mais efetiva?

Métricas Absolutas (Cenário Inicial)

À primeira vista, uma equipe parece ter encontrado muito mais problemas:

- **Equipe A:** **342 defeitos**
- **Equipe B:** **184 defeitos**

Ainda não sabemos!

Os projetos possuem tamanhos e complexidades muito diferentes:

Projeto A: 342 defeitos em **150 KLOC**

Projeto B: 184 defeitos em **50 KLOC**

Precisamos de uma métrica normalizada para comparar de forma justa.

Densidade de Defeitos

Normalização por tamanho do código (Defeitos / KLOC):

Equipe A (Projeto A)

2.28 defeitos / KLOC

Equipe B (Projeto B)

3.68 defeitos / KLOC

A **Equipe B** encontrou mais defeitos proporcionalmente ao tamanho do código, mostrando-se mais efetiva na identificação de falhas por volume desenvolvido.

Métricas Orientadas à Função

Medir o valor entregue ao usuário final em vez do esforço ou volume bruto de digitação.

Pontos de Função (PF)

Uma alternativa às linhas de código que foca na **funcionalidade oferecida ao usuário**.

Avalia a complexidade baseando-se em:

- Entradas, Saídas e Consultas
- Arquivos e Interfaces Externas

O paradoxo das linhas de código

Considere dois sistemas que resolvem o **mesmo problema**:

- **Sistema A:** 10.000 LOC
- **Sistema B:** 5.000 LOC

O sistema A necessariamente entregou mais valor? Não.

Métricas Orientadas à Função

Medir o valor entregue ao usuário final em vez do esforço ou volume bruto de digitação.

Pontos de Função (PF)

Uma alternativa às linhas de código que foca na **funcionalidade oferecida ao usuário**.

Avalia a complexidade baseando-se em:

- Entradas, Saídas e Consultas
- Arquivos e Interfaces Externas

O paradoxo das linhas de código

Considere dois sistemas que resolvem o **mesmo problema**:

- **Sistema A:** 10.000 LOC
- **Sistema B:** 5.000 LOC

O sistema A necessariamente entregou mais valor? Não.

Por que evitar a métrica LOC?

As linhas de código (**Lines of Code**) dependem diretamente de fatores técnicos e não do valor de negócio entregue:

- **Linguagem utilizada:** Abstrações diferentes resultam em volumes de código distintos.
- **Estilo e arquitetura:** Padrões de projeto e organização influenciam no tamanho físico dos arquivos.
- **Ferramentas e solução:** Geradores automáticos e a solução lógica adotada mudam drasticamente o esforço de digitação.

A métrica de função remove esse viés tecnológico.

Pontos de Função (PF): como calcular?

Pontos de Função medem o **tamanho funcional** do software pelas funcionalidades percebidas pelo usuário, sem depender de linguagem.

Função	O que representa	Simples	Média	Complexa
EE Entrada Externa	Dados que entram no sistema vindos de fora.	3	4	6
SE Saída Externa	Dados processados que saem do sistema para o usuário ou outros sistemas.	4	5	7
CE Consulta Externa	Consulta interativa de dados com entrada e saída diretas, sem cálculos complexos.	3	4	6
ALI Arq. Lógico Interno	Grupo de dados mantidos e controlados internamente dentro do sistema.	7	10	15
AIE Arq. Interface Externa	Dados apenas referenciados ou consumidos pelo sistema, mas mantidos por terceiros.	5	7	10

Multiplique quantidade × peso e some

PF = Σ (Quantidade de Funções × Peso)

Exemplo Prático:
- 5 EE simples → 5 × 3 = 15
- 2 SE médias → 2 × 5 = 10
- 1 ALI complexo → 1 × 15 = 15

Resultado: 15 + 10 + 15 = **40 PF**

Padrões Internacionais ISO

ISO/IEC 14143 Princípios fundamentais da Medição de Tamanho Funcional (FSM).
ISO/IEC 20926 Padronização do método de medição funcional **IFPUG**.

PF mede tamanho funcional, não esforço!
Dados de horas/PF ou R$/PF apoiam estimativas.

Métricas de Qualidade

No contexto de gerenciamento, ajudam a mensurar características essenciais:

- **Corretitude:** Entrega correta das funções.
- **Manutenibilidade:** Facilidade de alteração.
- **Usabilidade:** Experiência de uso.
- **Integridade:** Resistência a ataques.

Corretitude

Grau em que o software executa corretamente as funções exigidas. Medida baseada em falhas.

Métrica Tradicional

Defeitos / KLOC

Usabilidade

Mede a experiência e facilidade de operação:

- Facilidade de aprendizado.
- Tempo para eficiência no uso.
- Impacto na produtividade.
- Satisfação subjetiva.

Dica: Questionários com usuários complementam métricas quantitativas.

Manutenibilidade

Facilidade de correção, adaptação ou ampliação.

Tempo Médio para Mudança

Ciclo de atendimento:

1. Analisar a solicitação

Balanced Scorecard (BSC): Exemplo

Alinhando indicadores de desempenho de Engenharia de Software a objetivos estratégicos de negócio.

1. Perspectiva Financeira

Pergunta-chave: Como os stakeholders financeiros enxergam o projeto?

Métricas de Software: Retorno sobre Investimento (ROI), custo total de propriedade (TCO) e custo de atraso das entregas (**Cost of Delay**).

Foco: Garantir que as atividades de engenharia criem valor comercial real.

2. Perspectiva do Cliente

Pergunta-chave: Como os usuários finais e clientes percebem nosso produto?

Métricas de Software: Índice de Satisfação do Usuário, taxa de retenção, tempo de resposta de bugs reportados por clientes e facilidade de uso (**Usabilidade**).

Foco: Entregar valor real, qualidade e experiência de uso otimizada ao usuário.

3. Processos Internos de Negócio

Pergunta-chave: Em quais processos internos da engenharia devemos ser excelentes?

Métricas de Software: Densidade de defeitos, eficácia da revisão de código (**Code Review**), tempo médio de entrega (Lead/Cycle Time) e cobertura de testes.

Foco: Eficiência, previsibilidade de prazos e manutenibilidade do código.

4. Aprendizado e Crescimento

Pergunta-chave: Como podemos continuar melhorando e inovando?

Métricas de Software: Treinamentos concluídos por desenvolvedores, adoção de novas ferramentas/arquiteturas e satisfação/clima organizacional da equipe.

Foco: Desenvolvimento de competências, novas tecnologias e retenção de talentos.

Conclusão

Fundamentos

Medida
Quantifica um atributo físico ou lógico do ambiente.

Métrica
Relaciona diferentes medidas para fornecer contexto.

Indicador
Contextualiza métricas para gerar compreensão e apoiar decisões.

Medição
O processo estruturado para a obtenção das medidas.

Objetos de Medição

Processo
Como estamos desenvolvendo?
Foco na eficiência, metodologias e performance da equipe.

Projeto
Como está o trabalho?
Foco em cronogramas, custos, riscos e entregas da sprint.

Produto
Como está o software?
Foco em qualidade, manutenibilidade, corretitude e valor real.

Objetivos

Métricas não são um fim em si

1. Compreender
Enxergar o estado real do desenvolvimento.

2. Avaliar
Criticar desvios de esforço, qualidade e prazos.

3. Decidir
Direcionar ações corretivas baseadas em evidências.

4. Melhorar
Otimizar continuamente processos e produtos.

Referências

PRESSMAN, R. S.; MAXIM, B. R. Engenharia de Software: Uma Abordagem Profissional. 9. ed. Capítulos 23.

SOMMERVILLE, Ian. Engenharia de Software. 10. ed. São Paulo: Pearson, 2019.

IEEE COMPUTER SOCIETY. Guide to the Software Engineering Body of Knowledge — SWEBOK Guide. Version 4.0. IEEE Computer Society, 2024

Material Adicional

- Kazi, Lj, Biljana Radulovic, and Zoltan Kazi. "Performance indicators in software project monitoring: Balanced scorecard approach." 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics. IEEE, 2012.
- Hasan, Masum, et al. "Using a balanced scorecard to identify opportunities to improve code review effectiveness: An industrial experience report." Empirical Software Engineering 26.6 (2021): 129.
- Kaplan, Robert S., and David P. Norton. The balanced scorecard: measures that drive performance. Vol. 70. Boston, MA, USA: Harvard business review, 2005
- DORA's software delivery performance metrics: https://dora.dev/guides/dora-metrics/

Dúvidas e Discussão