Métricas de Software

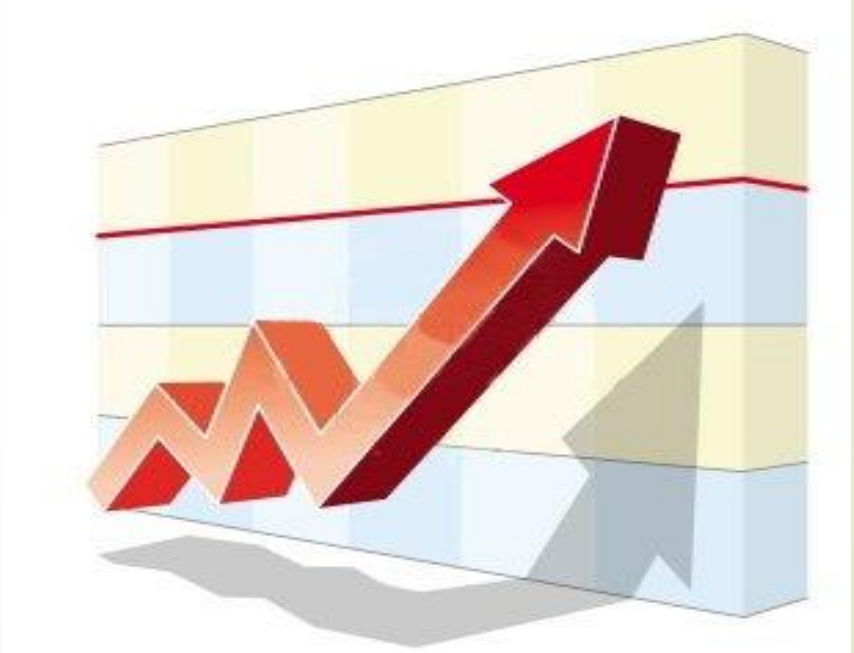
Prof. Wellington Moreira de Oliveira

Adaptado de Profa. Dra. Rosângela Aparecida Delloso Penteado

O que é métrica?

Muitos termos são usados, indistintamente, mas representam conceitos distintos.

- Medida
- Medição
- Métrica
- Indicador



Métricas Orientadas a Tamanho

Projeto	LOC	Esforço	$(000)	Pág. doc.	Erros	Defeitos	Pessoas
alfa	12.100	24	168	365	134	29	3
beta	27.200	62	440	1224	321	86	5
gama	20.200	43	314	1050	256	64	6
...	...	...	...	...	...	...	...

Erros por KLOC (milhares de linhas de código), **defeitos por KLOC**, **$ por KLOC** e **páginas de documentação por KLOC**.
Adicionalmente, outras métricas interessantes podem ser calculadas: **erros por pessoa-mês**, **KLOC por pessoa-mês**, **$ por página de documentação**.

Métricas Orientadas a Tamanho

Vantagens

- Fáceis de serem obtidas
- Vários modelos de medidas
- Baseados em LOC ou KLOC

Desvantagens

- LOC depende da linguagem de programação
- Penalizam programas bem projetados, mas pequenos
- Não se adaptam às linguagens não procedimentais
- Difícil de obter em fase de planejamento

Métricas Orientadas a Função

- Métricas de software orientadas a função usam uma **medida de funcionalidade** entregue pela aplicação como valor de normalização.
- A métrica orientada a função mais amplamente usada é a **pontos por função** (function point – FP).

Pontos por Função

Parâmetro de Medição	Contagem	Fator de Peso Simples	Fator de Peso Médio	Fator de Peso Complexo	Resultado Multiplicação
Entradas Externas (EIs)		x 3	x 4	x 6	
Saídas Externas (EOs)		x 4	x 5	x 7	
Consultas Externas (EQs)		x 3	x 4	x 6	
Arquivos Lógicos Internos (ILFs)		x 7	x 10	x 15	
Arquivos de Interface Externa (EIFs)		x 5	x 7	x 10	
Contagem Total	→				

Fatores de Ajuste - Fi

1. O sistema requer salvamento (backup) e recuperação (recovery)?
2. Comunicações de dados são necessárias?
3. Há funções de processamento distribuídas?
4. O desempenho é crítico?
5. O sistema vai ser executado em um ambiente operacional existente, intensamente utilizado?
6. O sistema requer entrada de dados on-line?
7. A entrada de dados on-line exige que a transação de entrada seja construída através de várias telas ou operações?
8. Os arquivos mestre são atualizados on-line?
9. As entradas, saídas, arquivos ou consultas são complexas?
10. O processamento interno é complexo?
11. O código é projetado para ser reusado?
12. A conversão e a instalação estão incluídas no projeto?
13. O sistema está projetado para instalações múltiplas em diferentes organizações?
14. A aplicação está projetada para facilitar modificações e para facilidade de uso pelo usuário?

Classificação das Fi

Cada resposta deve obedecer a uma escala de 0 a 5., em que 0 significa não-importante ou não-aplicável e 5 absolutamente essencial.

PF = total de contagem x [0,65 + 0,01 x Σ(Fi)]

Artigo para consulta

Albrecht, A. J. "Measuring Application Development Productivity". Proc. IBM Application Development Sysmposium, Monterey, CA, out. de 1979, p 83-92.

Métricas Orientadas a Função

Vantagens

- Independentes da linguagem;
- Ideal para aplicações que usam linguagem não procedimental;
- Se baseiam em dados mais fáceis de serem conhecidos durante a evolução do projeto.

Desvantagens

- Cálculo baseado em dados subjetivos;
- Resultado é apenas um número.

Linguagem de Programação	LOC por Ponto por Função			
	Média	Mediana	Baixa	Alta
Access	35	38	15	47
Ada	154	—	104	205
APS	86	83	20	184
ASP 69	62	—	32	127
Assembler	337	315	91	694
C	162	109	33	704
C++	66	53	29	178
Clipper	38	39	27	70
COBOL	77	77	14	400
Cool:Gen/IEF	38	31	10	180
Culprit	51	—	—	—
DBase IV	52	—	—	—
Easytrieve+	33	34	25	41
Excel47	46	—	31	63
Focus	43	42	32	56
FORTRAN	—	—	—	—
FoxPro	32	35	25	35
Ideal	66	52	34	203
IEF/Cool:Gen	38	31	10	180
Informix	42	31	24	57
Java	63	53	77	—
JavaScript	58	63	42	75
JCL	91	123	26	150
JSP	59	—	—	—
Lotus Notes	21	22	15	25
Mantis	71	27	22	250
Mapper	118	81	16	245
Natural	60	52	22	141

Pontos por caso de uso

A análise de sistemas Orientados a Objetos utiliza normalmente, os **diagramas de Casos de Uso** para **descrever as funcionalidades** do sistema.

Permite que seja possível estimar o tamanho do sistema ainda na **fase de levantamento de Casos de Uso**.

Grau de detalhe dos Casos de Uso analisados influencia diretamente na qualidade final da medição.

1. Calcular total de pesos não ajustados dos atores

Relacionar os atores, classificá-los de acordo com seu nível de complexidade (simples, médio ou complexo)

Complexidade do ator	Descrição	Peso
Simples	Muito poucas entidades de Banco de Dados envolvidas e sem regras de negócio complexas	1
Médio	Poucas entidades de Banco de Dados envolvidas e com algumas regras de negócio complexas	2
Complexo	Regras de negócios complexas e muitas entidades de Bancos de Dados presentes	3

TPNAA = 1*numAtoresSimples + 2*numAtoresMedio + 3*NumAtoresComplexo

2. Calcular pesos não ajustados dos casos de uso

Contar os casos de uso e atribuir o grau de complexidade sendo a complexidade com base no número de classes e transações.

Tipo de Caso de Uso	Descrição	Peso
Simples	Considerar até 3 transações com menos de 5 classes de análise	5
Médio	Considerar de 4 a 7 transações com 5 a 10 classes de análise	10
Complexo	Considerar de 7 transações com pelo menos de 10 classes de análise	15

TPNACU= 5*numCasoUsoSimples + 10*numCasoUsoMedio + 15*NumCasoUsoComplexo

3. Calcular pontos de casos de uso não ajustados

PCUNA = TPNAA + TPNACU

4. Calcular fator de complexidade técnica

Descrição	Peso
Sistemas Distribuídos	2,0
Desempenho da aplicação	1,0
Eficiência do usuário final (on-line)	1,0
Processamento interno complexo	1,0
Reusabilidade do código em outras aplicações	1,0
Facilidade de instalação	0,5
Usabilidade (facilidade operacional)	0,5
Portabilidade	2,0
Facilidade de manutenção	1,0
Concorrência	1,0
Características especiais de segurança	1,0
Acesso direto para terceiros	1,0
Facilidades especiais de treinamento	1,0

FCT = 0.6 + (0.01 * Somatório dos Ti*Peso)

5. Calcular fatores de complexidade ambiental

Fator	Descrição	Peso
F1	Familiaridade com o processo de desenvolvimento de software	1,5
F2	Experiência na aplicação	0,5
F3	Experiência com OO, na linguagem e na técnica de desenvolvimento	1,0
F4	Capacidade do líder de análise	0,5
F5	Motivação	1,0
F6	Requisitos estáveis	2,0
F7	Trabalhadores com dedicação parcial	-1,0
F8	Dificuldade da linguagem de programação	-1,0

FCA = 1.4 + (-0.03 * Somatório dos Fi*Peso)

6. Calcular pontos de casos de uso ajustados

PCUA = PCUNA * FCT * FCA

7. Calculos finais

- Pessoa-hora por unidade de PCU (Karner sugere 20)
- Estimativa em pessoa-hora (PCUA * PH-PCU)
- Tamanho da equipe (dado de entrada)
- Estimativa em horas(EPH/TE)
- Estimativa em meses (EH/160 - considerando 4 semanas e 40 horas por semana)

Artigo para consulta

Karner, G. "Resource Estimation for Objectory Projects". Objective Systems SF AB (copyright owned by Rational Software), 1993

Exercício PCU