UNIP – Tatuapé
Ciência da Computação / Sistemas de Informação
Engenharia de Software: Métricas de Software

COCOMO 81 - (COnstructive COst MOdel) – Modelo Construtivo de Custo

É um método que busca medir esforço, prazo, tamanho de equipe e custo necessário para o desenvolvimento do software, desde que se tenha a dimensão do mesmo, ou seja, pelo modelo de estimativa de tamanho de software com base histórica ou através do cálculo obtido pelo APF (Análise de Ponto de Função).

O modelo CoCoMo foi proposto por BARRY BOEHM (1981), tendo sido construído e calibrado inicialmente a partir de informação de, em torno de 83, projetos concluídos. Sua utilização tem permitido estimativas com um erro inferior a 20% em cerca de 70% dos projetos.

O CoCoMo 81 considera três modos de desenvolvimento:

- **Modo Orgânico:** é usado para calcular o esforço para um projeto onde as restrições de desenvolvimento são ligeiras; aplicável a ambientes de desenvolvimento estáveis, com pouca inovação e a projetos com equipes de dimensão relativamente pequena;

- **Modo Semidestacado:** é usado para um projeto em que as restrições sobre o projeto são superiores ao modo orgânico, mas resta ainda alguma flexibilidade; aplicável a projetos com características entre o modo orgânico e o embutido;

- **Modo Embutido:** é usado para um projeto que tem limitações definidas com muito rigor. O projeto como um todo é um pioneiro e, portanto, não pode confiar em projetos anteriores concluídos; aplicável no desenvolvimento de sistemas complexos embutidos em hardware, com muita inovação, com restrições severas e/ou com requisitos muito voláteis.

O CoCoMo considera ainda três "estágios" do modelo, à medida que vai sendo introduzido mais detalhamento:

- **Modelo Básico:** É um modelo estático de valor simples que computa o esforço e o custo de desenvolvimento de software como uma função do tamanho de programa expresso em linhas de código estimadas.

- **Modelo Intermediário:** A fase seguinte de sofisticação do modelo; corresponde a considerar a influência de um conjunto de vários fatores, relativos quer ao sistema a produzir (produto) propriamente dito, quer ao suporte computacional (tecnologia utilizada), fator humano e organização do processo de desenvolvimento de software. A influência destes fatores, em número de 15 no modelo originalmente proposto, deve ser avaliada numa escala discreta e ponderada.

- **Modelo Avançado:** Na sua versão mais completa, o modelo CoCoMo introduz aspectos adicionais como a decomposição de um sistema de grande dimensão em subsistemas. Outros aspectos correspondem à distribuição das estimativas de esforço e de prazo por fase e por atividade e à influência diferenciada de cada fator influenciador do custo por fase, ou seja, é a aplicação do COCOMO intermediário para cada fase do projeto.

UNIP – Tatuapé
Ciência da Computação / Sistemas de Informação
Engenharia de Software: Métricas de Software

O modelo de estimativa de custo CoCoMo é usado por uma grande quantidade de gerentes e baseado em um estudo de contagens de projetos de software. Ao contrário de outros modelos de estimativa de custo, o CoCoMo é um modelo aberto, todos os detalhes são incluídos como:

1. Equações subjacentes de estimativa de custo;
2. Suposições feitas no modelo;
3. Definições precisas da fase de projeto;
4. Custos incluídos de uma estimativa são indicados explicitamente.

As estimativas do CoCoMo são mais objetivas e repetíveis do que as feitas pelos métodos que confiam em modelos proprietários. Pode ainda refletir um ambiente de desenvolvimento do software e produzir estimativas bastante exatas.

Direcionadores de Custo

São fatores de ajuste do esforço, baseados em 15 atributos direcionadores do custo que estão divididos em quatro categorias (produto, computador, pessoal e projeto).

Cada direcionador de custo é estimado com base em uma escala de seis pontos, variando de baixa para alta importância, conforme a Tabela 1. O produto de todos os multiplicadores de esforço é o Fator de Ajustamento de Esforço ou EAF (*Effort Adjustment Factor*).

Tabela 1 – Direcionadores de Custo CoCoMo 81

Direcionador de custo	***Descrição***	***Muito Baixo***	***Baixo***	***Nominal***	***Alto***	***Muito Alto***	***Extra Alto***
Produto							
RELY	Confiabilidade do software	0,75	0,88	1,00	1,15	1,40	-
DATA	Tamanho do banco de dados	-	0,94	1,00	1,08	1,16	-
CPLX	Complexidade do produto	0,70	0,85	1,00	1,15	1,30	1,65
Computador							
TIME	Restrição de tempo de execução	-	-	1,00	1,11	1,30	1,66
STOR	Restrição memória principal	-	-	1,00	1,06	1,21	1,56
VIRT	Mudanças do ambiente	-	0,87	1,00	1,15	1,30	-
TURN	Velocidade processamento computador	-	0,87	1,00	1,07	1,15	-
Pessoal							
ACAP	Capacidade do analista	1,46	1,19	1,00	0,86	0,71	-
AEXP	Experiência na aplicação	1,29	1,13	1,00	0,91	0,82	-
PCAP	Capacidade do programador	1,42	1,17	1,00	0,86	0,70	-
VEXP	Experiência com hardware	1,21	1,10	1,00	0,90	-	-
LEXP	Experiência na linguagem	1,14	1,07	1,00	0,95	-	-
Projeto							
MODP	Práticas modernas de programação	1,24	1,1	1,00	0,91	0,82	-
TOOL	Ferramentas de software	1,24	1,10	1,00	0,91	0,83	-
SCED	Cronograma de desenvolvimento	1,23	1,08	1,00	1,04	1,10	-

Fonte: BOEHM (1981)

UNIP – Tatuapé
Ciência da Computação / Sistemas de Informação
Engenharia de Software: Métricas de Software

COCOMO Básico

É um modelo estático que calcula o esforço de desenvolvimento de software e seu custo, em função do tamanho de linhas de códigos desenvolvidas.

$E = ab(KLOC)^{bb}$
$D = cb(E)^{db}$
$P = E / D$

Onde:

- E é o esforço aplicado pela pessoa no mês;
- D é o tempo de desenvolvimento em meses cronológicos;
- KLOC é o número calculado de linhas de código para o projeto (expressado em milhares);
- P é o número das pessoas necessário.

Os coeficientes ab, bb, cb e db são dados na seguinte tabela:

Projeto de Software	**ab**	**bb**	**cb**	**db**
Orgânico	2,4	1,05	2,5	0,38
Semidestacado	3,0	1,12	2,5	0,35
Embutido	3,6	1,20	2,5	0,32

Cocomo básico é bom por ser rápido em estimativas e custos de software, mas sua exatidão é limitada por causa de sua falta de fatores para explicar as diferenças entre ferramentas, qualidade de pessoal e experiência, uso de ferramentas modernas e técnicas, e outros atributos de projeto que influenciam nos custos de software.

Estudo de Caso: Projeto A-380 (COCOMO – Básico)

Para o projeto do A-380, o maior avião do mundo atualmente, o centro de tecnologia Aeronáutica da AirBus, estimou que o software de tempo real para esta aeronave teria aproximadamente 360.000 linhas de código. Diante da complexidade do software e seus requisitos extremamente particulares, responda as seguintes questões:

I) Determine o esforço, segundo o COCOMO – Básico;
II) Calcule o tempo necessário para a realização do projeto;
III) Estime o número de pessoas necessárias.

Fórmulas Possíveis:

Modos	Esforço (Pessoas/Mês)	Tempo (Mês)
Orgânico	$Esforço = 2,4 * KLOC^{1,05}$	$D = 2,5 * Esforço^{0,38}$
Semidestacado	$Esforço = 3,0 * KLOC^{1,12}$	$D = 2,5 * Esforço^{0,35}$
Embutido	$Esforço = 3,6 * KLOC^{1,20}$	$D = 2,5 * Esforço^{0,32}$

Interpretando o texto, é possível observar que trata-se de um sistema complexo e de grande porte. Portanto o modelo correto é o Embutido.

$Esforço = 3,6 * 360^{1,20} =$ 4205,9635 Pessoas/Mês
$Tempo = 2,5 * 4205,96^{0,32} =$ 36,1058 Meses
Pessoas = 4205,9635 / 36,1058 = 116,4900 Pessoas

Prof. Marcelo Nogueira

UNIP – Tatuapé
Ciência da Computação / Sistemas de Informação
Engenharia de Software: Métricas de Software

COCOMO Intermediário

Calcula o esforço de desenvolvimento de software em função do tamanho do programa, que inclui custo, avaliação subjetiva do produto, ferramentas, pessoal e atributos de projeto.

$E = ai * (KLOC)^{(bi)} * EAF$
$D = ci(E)^{di}$
$P = E / D$

Onde:

- E é o esforço aplicado em pessoas por mês;
- KLOC é o número de linhas de código para o projeto;
- EAF é o fator calculado a partir dos direcionadores de custo.
- D é o tempo de desenvolvimento em meses cronológicos;
- P é o número das pessoas necessário.

Os coeficientes ai, bi, ci e di são dados na próxima tabela:

Projeto de Software	ai	bi	ci	di
Orgânico	3,2	1,05	2,5	0,38
Semidestacado	3,0	1,12	2,5	0,35
Embutido	2,8	1,20	2,5	0,32

O método intermediário é uma extensão do método básico, mas com mais categorias de controle como: Atributos do produto, Atributos de hardware, Atributos pessoais, Atributos do projeto.

Já os direcionadores de custo são dados na próxima tabela:

Direcionador de custo	***Descrição***	***Muito Baixo***	***Baixo***	***Nominal***	***Alto***	***Muito Alto***	***Extra Alto***
Produto							
RELY	Confiabilidade do software	0,75	0,88	1,00	1,15	1,40	-
DATA	Tamanho do banco de dados	-	0,94	1,00	1,08	1,16	-
CPLX	Complexidade do produto	0,70	0,85	1,00	1,15	1,30	1,65
Computador							
TIME	Restrição de tempo de execução	-	-	1,00	1,11	1,30	1,66
STOR	Restrição memória principal	-	-	1,00	1,06	1,21	1,56
VIRT	Mudanças do ambiente	-	0,87	1,00	1,15	1,30	-
TURN	Velocidade processamento computador	-	0,87	1,00	1,07	1,15	-
Pessoal							
ACAP	Capacidade do analista	1,46	1,19	1,00	0,86	0,71	-
AEXP	Experiência na aplicação	1,29	1,13	1,00	0,91	0,82	-
PCAP	Capacidade do programador	1,42	1,17	1,00	0,86	0,70	-
VEXP	Experiência com hardware	1,21	1,10	1,00	0,90	-	-
LEXP	Experiência na linguagem	1,14	1,07	1,00	0,95	-	-
Projeto							
MODP	Práticas modernas de programação	1,24	1,1	1,00	0,91	0,82	-
TOOL	Ferramentas de software	1,24	1,10	1,00	0,91	0,83	-
SCED	Cronograma de desenvolvimento	1,23	1,08	1,00	1,04	1,10	-

UNIP – Tatuapé
Ciência da Computação / Sistemas de Informação
Engenharia de Software: Métricas de Software

Estudo de Caso: Projeto A-380 (COCOMO – Intermediário)

Como descrito anteriormente, para o projeto do A-380, o maior avião do mundo atualmente, o centro de tecnologia Aeronáutica da AirBus, estimou que o software de tempo real para esta aeronave teria aproximadamente 360.000 linhas de código. Diante da complexidade do software e seus requisitos extremamente particulares, responda as seguintes questões:

I) Determine o esforço, segundo o COCOMO – Intermediário;
II) Calcule o tempo necessário para a realização do projeto;
III) Estime o número de pessoas necessárias.

Passo 1: Tamanho do Produto de Software: O modelo utiliza um tamanho de produto expresso em fontes de linha de código (KLOC). Quanto maior o tamanho maior o esforço e o prazo.

KLOC = 360.

Passo 2: Selecionar complexidade do projeto de software:

Projeto de Software	ai	bi	ci	di
Orgânico	3,2	1,05	2,5	0,38
Semidestacado	3,0	1,12	2,5	0,35
Embutido	2,8	1,20	2,5	0,32

Passo3: Selecione o Direcionador de Custo (EAF):

Direcionador de custo	***Descrição***	***Muito Baixo***	***Baixo***	***Nominal***	***Alto***	***Muito Alto***	***Extra Alto***
Produto							
RELY	Confiabilidade do software	0,75	0,88	1,00	1,15	1,40	-
DATA	Tamanho do banco de dados	-	0,94	1,00	1,08	1,16	-
CPLX	Complexidade do produto	0,70	0,85	1,00	1,15	1,30	1,65
Computador							
TIME	Restrição de tempo de execução	-	-	1,00	1,11	1,30	1,66
STOR	Restrição memória principal	-	-	1,00	1,06	1,21	1,56
VIRT	Mudanças do ambiente	-	0,87	1,00	1,15	1,30	-
TURN	Velocidade processamento computador	-	0,87	1,00	1,07	1,15	-
Pessoal							
ACAP	Capacidade do analista	1,46	1,19	1,00	0,86	0,71	-
AEXP	Experiência na aplicação	1,29	1,13	1,00	0,91	0,82	-
PCAP	Capacidade do programador	1,42	1,17	1,00	0,86	0,70	-
VEXP	Experiência com hardware	1,21	1,10	1,00	0,90	-	-
LEXP	Experiência na linguagem	1,14	1,07	1,00	0,95	-	-
Projeto							
MODP	Práticas modernas de programação	1,24	1,1	1,00	0,91	0,82	-
TOOL	Ferramentas de software	1,24	1,10	1,00	0,91	0,83	-
SCED	Cronograma de desenvolvimento	1,23	1,08	1,00	1,04	1,10	-

Passo 4: Realizar os cálculos:

EAF = (Produto de todos os direcionadores de custo) = 1,4470328517

E = ai * (KLOC)$^{(bi)}$ * EAF
E = 2,8 * 360 ^{1,20} * 1,4470328517 = 4733,68574647 (Pessoas / Mês)

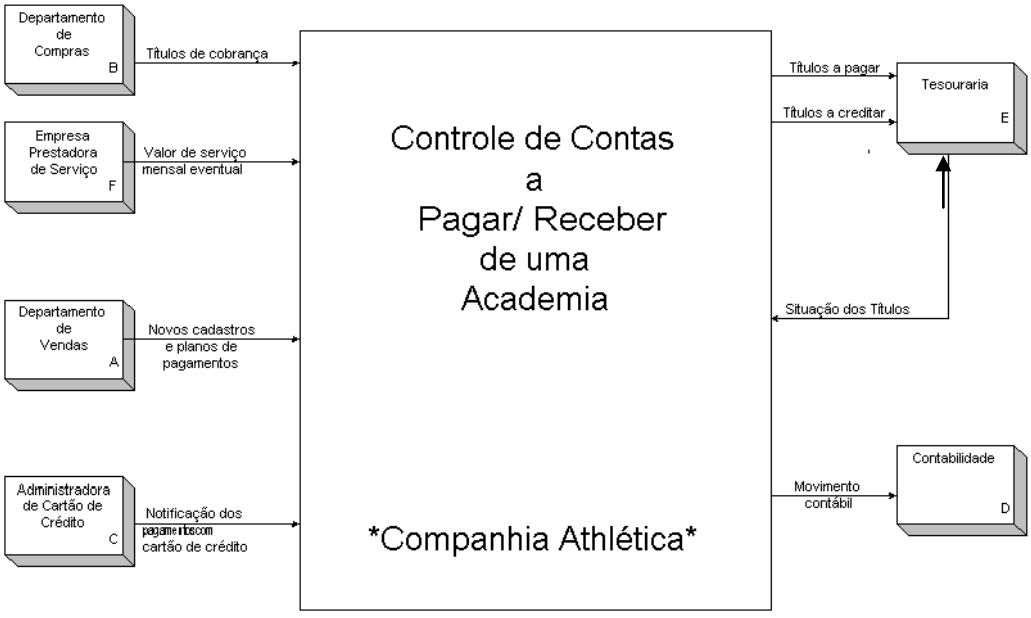
D = ci(E)^{di}
D = 2,5 * 4733,68574647 ^{0,32} = 37,497630401 (Meses)

P = E / D
P = 4733,68574647 / 37,497630401 = 126,239596898 (Pessoas)

KLOC

Quando não há estimativa de quantidade de linhas de código, é possível calcular através da análise de ponto de função (APF). Assim segue:

Para tanto, segue o diagrama de contexto de um sistema de controle de contas a pagar / receber de uma academia.



UNIP – Tatuapé
Ciência da Computação / Sistemas de Informação
Engenharia de Software: Métricas de Software

Passo 1: Calcular a complexidade do projeto:

Medição Parâmetro	QT		Complexidade Baixa	Complexidade Média	Complexidade Alta		Total
Entradas (E)	04	X	**3**	**4**	**6**	=	16
Saídas (S)	03	X	**4**	**5**	**7**	=	21
Consultas (C)	01	X	**3**	**4**	**6**	=	04
Arquivos (A)	05	X	**7**	**10**	**15**	=	35
Interfaces Externas (IE)	01	X	**5**	**7**	**10**	=	07
Total Contagem							83

Passo 2: Ajustar a complexidade:

Os 14 fatores de ponderação de complexidade (F_i) - Taxa de cada fator numa escala de 0 a 5: (0 = Sem influência, 1 = incidental, 2 = moderado, 3 = médio, 4 = significativas, 5 = Essencial):

Nº	Perguntas	Fi
1	O sistema requer salvamento (*backup*) e recuperação (*recovery*)?	3
2	Comunicações de dados são necessárias?	4
3	Há funções de processamentos distribuídos?	5
4	O desempenho é crítico?	2
5	O sistema vai ser executado em um ambiente operacional existente, intensamente utilizado?	1
6	O sistema requer entrada de dados on-line?	5
7	A entrada de dados on-line exige que a transação de entrada seja construída através de várias telas ou operações?	3
8	Os arquivos mestre são utilizados on-line?	2
9	As entradas, saídas, arquivos ou consultas são complexas?	4
10	O processamento interno é complexo?	3
11	O código é projetado para ser reusado?	3
12	A conversão e a instalação estão incluídas no projeto?	5
13	O sistema está projetado para instalações múltiplas em diferentes organizações?	0
14	A aplicação está projetada para facilitar modificações e para facilidade de uso pelo usuário?	3
	Total	43

Passo 3: Calcular a quantidade de pontos de função:

FP = total de contagem x $[0{,}65 + 0{,}01 \times \Sigma(F_i)]$

FP = 83 * [0,65 + 0,01 * 43] =
FP = 83 * [0,65 + 0,43] =
FP = 83 * [1,08] =
FP = 89,64

Prof. Marcelo Nogueira

Passo 4: Encontrar LOC (linhas de código), escolhendo uma linguagem de programação que você irá usar ao desenvolver um projeto:

Language	Default SLOC / UFP	Language	Default SLOC / UFP
Access	38	Jovial	107
Ada 83	71	Lisp	64
Ada 95	49	Machine Code	640
AI Shell	49	Modula 2	80
APL	32	Pascal	91
Assembly - Basic	320	PERL	27
Assembly - Macro	213	PowerBuilder	16
Basic - ANSI	64	Prolog	64
Basic - Compiled	91	Query – Default	13
Basic - Visual	32	Report Generator	80
C	128	Second Generation Language	107
C++	55	Simulation – Default	46
Cobol (ANSI 85)	91	Spreadsheet	6
Database – Default	40	Third Generation Language	80
Fifth Generation Language	4	Unix Shell Scripts	107
First Generation Language	320	USR_1	1
Forth	64	USR_2	1
Fortran 77	107	USR_3	1
Fortran 95	71	USR_4	1
Fourth Generation Language	20	USR_5	1
High Level Language	64	Visual Basic 5.0	29
HTML 3.0	15	Visual C++	34
Java	53		

LOC = Média (LOC / FP) X FP = LOC
LOC = (C++ = 55) X 89,64 = 4930,2
KLOC = 4930,2 / 1000 = 4,9302

UNIP – Tatuapé
Ciência da Computação / Sistemas de Informação
Engenharia de Software: Métricas de Software

ANEXO: Boehm, Barry "Software Engineering Economics", Prentice Hall 1981
TABLE 8-3 Software Cost Driver Ratings, Page 119

Ratings	Cost Drivers					
	Very Low	Low	Nominal	High	Very High	Extra High
Code	VL	LO	NM	HI	VH	XH
RELY	Effect: slight in convenience	Low, easily recoverable losses	Moderate, recover able losses	High financial loss	Risk to human life	
DATA	No Data	Text or INI Based Data	DB	DB with complex relationships and or overlapping views	DB with complex relationships and or overlapping views in < interactive time	
CPLX	Single Form Apps Obvious Functions (control Panel)	Single Form Apps complex functions (Utilities)	Multiple forms single threaded, single user	Multiple forms, multi threaded or multi-user	Both threaded and multiuser with large complex functionality or embedded in or used to control hardware	Critical software control systems like O/S.
TIME			<50% use of CPU	70%	85%	95%
STOR			<50% use of DISK or disk storage is not a factor	70%, disk space limited	85%, disk space limited	95%
VIRT		Major change every 12 months, minor: 1 month interactive	Major: 6 mos Minor: 2 weeks	Major: 2 mosMinor: 1 Week	Major: 2 weeks Minor: 2 days	
TURN		Interactive	<4hrs	4-12 Hours	>12 hours	
ACAP	15th %	35th %	55th %	75th %	90th %	
AEXP	< 4 months	1 year	3 years	6 years	12 years	
PCAP	15th %	35th %	55th %	75th %	90th %	
VEXP	< 1 MO	4 months	1 year	3 years		
LEXP	< 1 MO	4 months	1 year	3 years		
MODP	No Use	Beginning Use	Some Use	General Use	Routine Use	
TOOL	Little Tools	2 GL	3 GL with full IDE and Debugging facilities	4GL or 3+GL with component or object templates or premade component pieces	Additional Project Management Tools	
SCHED	70% Nom	85%	100%	130%	160%	

REFERÊNCIAS BIBLIOGRÁFICAS

LÓPEZ, Pablo A.P. "COCOMO II - Um modelo para estimativa de custos de Projetos de Software". Universidade Vale do Rio dos Sinos - UNISINOS, São Leopoldo - RS.
http://www.cin.ufpe.br/~mgcasf/Qualidade/TC_PabloArieldoPradoLopez.pdf

LÓPEZ, Pablo A.P. "COCOMO II - Um modelo para estimativa de custos de Gerência de Projetos". Universidade Vale do Rio dos Sinos - UNISINOS, São Leopoldo - RS.
http://www.cin.ufpe.br/~mgcasf/Qualidade/RA_PabloArielDoPadroLopez.pdf

Basic COCOMO Model
http://www.engin.umd.umich.edu/CIS/course.des/cis525/js/f00/gamel/cocomo.html

Método Cocomo
http://pt.wikipedia.org/wiki/M%C3%A9todo_COCOMO

Cocomo Reference Manual
http://sunset.usc.edu/research/COCOMOII/cocomo81docs/cocomo.pdf