

Estimativas de Software - Fundamentos, Técnicas e Modelos – Parte 2

O COCOMOII e uma introdução ao seu processo de calibração



Carlos Eduardo Vazquez

Mini currículo: Sócio-fundador da FATTO Consultoria e Sistemas, um dos autores do livro "Análise de Pontos de Função: Medição, Estimativas e Gerenciamento de Projetos de Software", atualmente em sua 8ª edição. Pioneiro na aplicação de métricas de software no Brasil possui 20 anos de experiência em TI, notoriamente na aplicação das disciplinas do desenvolvimento e sustentação de sistemas corporativos. Graduado em Processamento de Dados pela PUC-RJ em 1990, já passou com sucesso por quatro vezes pelo processo de certificação de especialista em pontos de função pelo IFPUG - International Function Point Users Group, tendo sido um dos primeiros brasileiros a conquistar essa certificação em 1996. Desde 1993, vem formando profissionais na aplicação da Análise de Pontos de Função, tendo sido professor da UFES, atuado como consultor de grandes projetos de tecnologia em empresas do setor financeiro, bancário e de telecomunicações.

De que se trata o artigo?

Estimar é uma atividade cotidiana, sistematicamente evitada por aqueles responsáveis pela sua execução. Na busca por superar isso, uma série de técnicas e ferramentas surge no cenário do desenvolvimento e manutenção de sistemas. Muitas vezes, no desespero por uma solução imediata, são adotadas independentemente de sua adequação ao cenário específico em que serão introduzidas ou mesmo apenas com um conhecimento superficial quanto ao seu funcionamento. Ferramentas como o COCOMOII, Simulação de Monte Carlo e Pontos de Função não substituem a analista responsável pela estimativa, que enfrenta a confusão entre diferentes atos, entre o que seja uma estimativa técnica, um compromisso pessoal ou uma meta corporativa.

Para que serve?

Nosso objetivo é diferenciar entre esses diferentes atos e como se portar diante de cada

um deles; destacar que simples cuidados podem ajudar a produzir estimativas de mais qualidade; apresentar como funciona uma série de ferramentas isoladamente e como integrá-las no estabelecimento de um ambiente propício à melhoria contínua da qualidade das estimativas.

Em que situação o tema é útil?

Nas diferentes situações em que um analista deve se relacionar com seus clientes no sentido de fornecer a sua expectativa para um prazo, custo, esforço ou escopo no desenvolvimento e manutenção de software. Visa ajudar a esse analista a identificar os diferentes tipos de solicitação e evitar que ele caia em armadilhas que o leve a assumir compromissos inexecutáveis. Adicionalmente, é útil também àquele profissional que trabalha na definição de processos de desenvolvimento e seleção de métodos e ferramentas para fins de melhorar o processo de estimativa de sua organização.

No artigo da edição anterior, os fundamentos para estimativas profissionais foram apresentados; algumas técnicas de estimativa exploradas, ainda que superficialmente; e foi feita a introdução sobre o COCOMOII. O objetivo deste texto é dar seqüência na exploração desse modelo e conhecer os passos comuns necessários à realização de qualquer estudo de produtividade e da aplicação dos seus resultados no planejamento de projetos de software. Nesse processo, vários dos conceitos básicos e gerais apresentados na edição anterior se mostram úteis e aplicados ao modelo de estimativas em questão.

A definição das fases da produção de software numa perspectiva gerencial e dos marcos que as delimitam

Um dos primeiros desafios ao conceber um modelo de custos para a engenharia de software é estabelecer um conjunto de premissas que permitam utilizar o modelo consistentemente entre diferentes organizações de software, usando diferentes abordagens e estratégias para entregar os produtos de software demandados pelos seus clientes.

O COCOMOII define e estabelece uma série de premissas que viabilizam esse objetivo. Elas também permitem que o modelo seja de muito valor na realização de estudos de produtividade na medida em que facilita a normalização na elaboração de estimativas (dados previstos), e a subsequente análise do que foi realizado (dados realizados).

O primeiro passo nessa exploração do COCOMOII é construir um conhecimento daquilo que o modelo define e estabelece como premissa. Nesse processo, a divisão do projeto em fases (não necessariamente em disciplinas como análise de requisitos, modelagem, codificação e testes, por exemplo) é o primeiro passo nessa construção.

As fases devem ser externas à função de desenvolvimento e manutenção de software de tal forma que possam ser usadas para fins de planejamento de alto nível, quando ainda se tem poucos detalhes sobre o objeto do trabalho e os riscos de escopo e produtividade associados ao mesmo ainda são muitos. Os responsáveis pela elaboração do COCOMOII não têm a intenção de revolucionar no que diz respeito a essa definição nas fases do desenvolvimento de software e buscam usar o que é geralmente aceito pela comunidade de engenharia de software: a estratégia de desenvolvimento em cascata e a abordagem denominada de processo unificado.

A Figura 1 é chave para o entendimento do COCOMOII e será mais explorada na medida em que for sendo construído o conhecimento sobre o modelo. Neste ponto, ela ilustra os marcos entre as fases das diferentes estratégias de desenvolvimento citadas.

As fases de iniciação, elaboração, construção e transição são aquelas definidas pelo Rational Unified Process (RUP), enquanto as fases de planos e requisitos, projeto preliminar, projeto detalhado, codificação e testes de unidade, testes e integração são aquelas encontradas em projetos utilizando um ciclo de vida em cascata (waterfall). O COCOMOII identifica e posiciona marcos comuns entre essas duas diferentes estratégias de desenvolvimento citadas, esse nivelamento é estabelecido com base nos produtos que se espera receber do projeto nesses marcos. Isso permite: (a) que sejam feitas estimativas por fase, o que facilita o planejamento na medida em que as diferentes fases têm maior intensidade de determinados tipos de trabalho com diferentes perfis de recursos humanos necessários; (b) que o responsável pela estimativa se localize no tempo, no momento do projeto e,

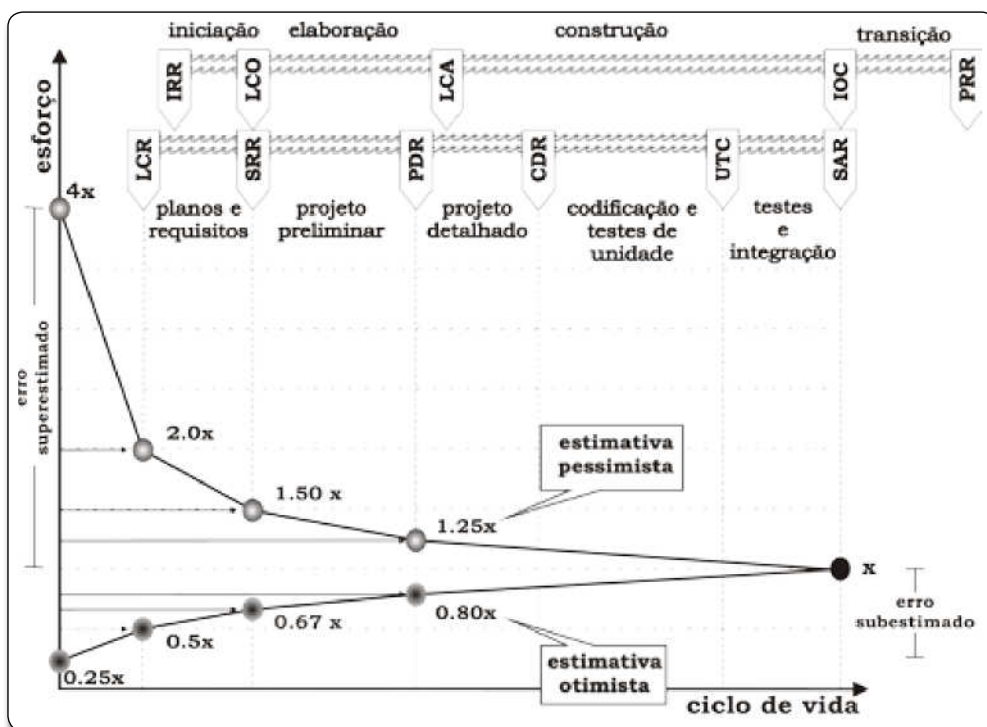


Figura 1. Fases do ciclo de vida e a sua integração com o COCOMOII

Marcos para Gestão do Processo	
Waterfall	RUP
LCR – Revisão de Conceito do Ciclo de Vida (<i>Life Cycle Concept Review</i>)	IRR – Revisão de Prontidão de Concepção
SRR – Revisão de Requisitos do Software (<i>Software Requirements Review</i>)	LCO – Revisão de Objetivos do Ciclo de Vida (<i>Life Cycle Objectives Review</i>)
PDR – Revisão de Projeto do Produto (<i>Product Design Review</i>)	LCA – Revisão da Arquitetura do Ciclo de Vida (<i>Life Cycle Architecture Review</i>)
CDR – Revisão de Projeto Crítico (<i>Critical Design Review</i>)	IOC – Capacidade Operacional Inicial (<i>Initial Operational Capability</i>)
UTC – Critério de Teste de Unidade (<i>Unit Test criteria</i>)	PRR – Revisão de Liberação do Produto (<i>Product Release Review</i>)
SAR – Revisão de Aceitação do Software (<i>Software Acceptance Review</i>)	

Tabela 1. Marcos conforme a estratégia de processo de desenvolvimento para fins de gestão

com isso, possa estabelecer o quanto já deveria ter sido feito e o quanto resta a fazer; e (c) que haja condições para uma análise de produtividade econômica (considerado como um requisito necessário para estimativas de qualidade).

A **Tabela 1** descreve os acrônimos dos marcos apresentados na **Figura 1**. A definição desses marcos não é do COCOMOII, que buscou usar padrões reconhecidos e estabelecidos, estando sua qualificação completa fora do escopo deste texto.

Na experiência do autor deste texto, ainda há na comunidade de profissionais de TI brasileira uma certa dificuldade em compreender a dinâmica nos processos de desenvolvimento iterativos e incrementais. Muitos utilizam o vocabulário e a terminologia do RUP, porém trabalham na verdade com um desenvolvimento em cascata.

De forma simplificada, estratégias de desenvolvimento baseadas no processo unificado buscam a realização de atividades de levantamento de requisitos em determinados pacotes de trabalho enquanto já há codificação ou modelagem de outros.

Observar o processo unificado não impede que haja fases definidas para fins de gerenciamento externo. Independentemente da estratégia de desenvolvimento utilizada, os clientes, aqueles que patrocinam o empreendimento, esperam uma série de produtos por eles reconhecidos e aceitos conforme o ciclo de vida do desenvolvimento evolui. Alguns desses produtos são definidos entre esses últimos e os responsáveis pela função de desenvolvimento no âmbito do próprio projeto já em execução e com base em princípios previamente definidos.

Por exemplo, uma empresa atuante em todo o país é responsável pela evolução de seu produto de gestão de recursos humanos em janelas sucessivas de três meses (time box). Nesse cenário, é a quantidade e tamanho das demandas que se ajusta ao prazo, e não o contrário. Procura-se atender ao maior número possível de demandas de melhorias em cada versão com um estoque de trabalho essencialmente constante. Quem cumpre o papel de demandante da função de desenvolvimento é a função de planejamento, que pode ser vista como um preposto interno, do cliente externo.

A unidade de gestão originalmente utilizada no processo de atendimento das demandas é o resultado da ação de um arquiteto de software, técnico, e não tem associado a ela um significado para os profissionais da função de planejamento. Essa unidade de gestão torna-se um pacote de trabalho que passa por diferentes etapas durante o seu atendimento na

função de desenvolvimento. O processamento desse pacote de trabalho segue uma estratégia essencialmente em cascata. As etapas existentes para fins de acompanhamento do progresso são relativas a cada pacote de trabalho individualmente, e não para o projeto como um todo.

Em outras palavras, o cliente não tem condição alguma de acompanhar o seu atendimento e fica numa posição em que tem que confiar que a função de desenvolvimento atingirá os objetivos estabelecidos em comum. Esse cenário impede uma gestão externa à função de desenvolvimento. Falta uma unidade de gestão que: (a) tenha significado para a função de planejamento; (b) possa ser medida de forma relevante à função de planejamento; (c) apóie o planejamento e acompanhamento das fases relativas à execução do projeto como um todo.

Para mudar essa situação, foi estabelecida uma unidade de gestão com significado para o departamento de planejamento que cumpre o papel de uma Ordem de Serviço, e foram definidas fases para o acompanhamento do projeto como um todo. São elas: Negociação/Planejamento da Versão; Concepção; Elaboração; Construção; e Transição. A fase de Negociação/Planejamento de Versão compreende:

a) Estimativas de escopo e esforço – elaboração do cronograma macro considerando um prazo total de três meses para as fases de concepção, elaboração, construção e transição usando os percentuais do prazo em cada fase (**Tabela 2**).

	Esforço	Prazo
Negociação / Planejamento	8,54%	50,00%
Concepção	11,56%	13,33%
Elaboração	12,70%	66,67%
Construção	64,36%	66,67%
Transição	2,85%	16,67%

Tabela 2. Distribuição do esforço e prazo.

Os percentuais descritos na **Tabela 2** foram obtidos por aproximação do esforço gasto no cenário atual onde não há utilização de fases externas e servem de referência preliminar. Na medida em que houver a institucionalização das mudanças propostas, esses percentuais devem ser revistos e gradualmente cumprirão um papel mais importante na definição do cronograma macro citado.

Marco	Objetivos do Ciclo de Vida (LCO)	Arquitetura do Ciclo de Vida (LCA)
Definição do conceito de operação	Objetivos e escopo de alto nível da aplicação; fronteira entre aplicações; premissas e parâmetros do ambiente; carências atuais do sistema; conceito de operação; cenários padrões, papéis e responsabilidades dos stakeholders.	Elaboração dos objetivos e escopo do sistema por incremento; elaboração do conceito de operação por incremento; cenários padrões e de exceção chave.
Protótipo(s) do sistema	Simulação com o uso de cenários; resolução dos riscos críticos.	Simulação de uma gama de cenários de uso; resolução dos principais riscos pendentes.
Definição de requisitos de sistema e software	Recursos de alto nível, interfaces, atributos e níveis de qualidade, incluindo: Evolução dos requisitos; prioridades; e concordância dos stakeholders no essencial. (Observe que essencial é uma resolução tomada antes do início da concepção/iniciação).	Elaboração de funções, interfaces, atributos de qualidade por incremento; Identificação de itens a serem definidos; evolução dos requisitos; concordância dos stakeholders em suas preocupações prioritárias.
Definição da arquitetura do sistema e do software	Definição de alto nível de pelo menos uma arquitetura possível; elementos e relacionamentos físicos e lógicos; escolhas de "software de prateleira" e componentes reutilizáveis de software; identificação de opções de arquitetura inviáveis.	Escolha de arquitetura e elaboração por incremento; restrições, configurações, conexões, componentes lógicos e físicos; Escolha de "software de prateleira", componentes reutilizáveis; escolha de estilo e domínio de arquitetura; parâmetros de evolução da arquitetura.
Definição do plano do ciclo de vida	Identificação dos stakeholders do ciclo de vida: Usuários, clientes, desenvolvedores, mantenedores, integradores, público em geral, outros; identificação do modelo de processo para o ciclo de vida; estágios de alto nível, incrementos; WWWWWHH de alto nível por estágio. (WWWWWHH – Por que (Why), o que (What), quando (When), quem (Who), onde (Where), como (How), quanto (How Much).	Elaboração de WWWWWHH para o início da capacidade operacional; identificação e elaboração parcial dos itens a serem definidos para os incrementos subsequentes.
Racional de Viabilidade	Garantia de consistência entre os elementos acima pela análise, medição, prototipação, simulação, dentre outros, não cabendo a este texto esgotar as providências para esse fim; análise de casos de negócio para os requisitos, arquiteturas possíveis.	Garantia de consistência entre os elementos acima; racional das principais opções rejeitadas; todos os principais riscos resolvidos ou cobertos pelo plano de gestão de riscos compreendido no plano do ciclo de vida.

Tabela 3. Produtos esperados conforme o marco e a fase do ciclo e vida

b) Definição, para cada ordem de serviço, de quais serão os produtos esperados para que a fase de Elaboração seja considerada concluída. A política é que a especificação funcional das ordens de serviço mais críticas esteja concluída e os riscos arquiteturais prioritários estejam resolvidos. Nesse ponto, espera-se que até 40% do prazo tenha se passado e até 25% do esforço consumido.

c) Marco de término: conjunto com todas as ordens de serviço da versão com o escopo validado e a primeira planilha de contagem de PF por produto.

d) Meta para término: duas semanas antes do final da construção da versão anterior.

A fase de Concepção compreende:

a) Atividades de um comitê para controle de alterações envolvendo o analista de requisitos, arquiteto e analista de negócio onde é feita uma nova validação funcional mais cuidadosa, avaliação do impacto da ordem de serviço no produto e a elaboração das unidades de gestão internas à função de desenvolvimento.

b) Revisão das estimativas de escopo e prazo.

c) Marco de término: Realização e finalização dos comitês para controle de alterações referentes a todas as ordens de serviço originalmente programadas para atendimento no projeto da versão.

d) Meta para término: Final da fase de Construção do projeto da versão anterior.

O exemplo exposto procurou ilustrar os conceitos apresentados referentes à definição de fases externas à função

de desenvolvimento para uma organização em particular e em termos de uma experiência prática, no mundo real. Esses marcos apresentados foram definidos a partir das premissas estabelecidas no COCOMOII e expostos na **Tabela 3**, especificamente entre as fases de Iniciação, Elaboração e Construção.

As fases cobertas pela estimativa do COCOMOII

Existem diversas ferramentas, sites, planilhas e até scripts de awk, como o disponível em <http://web.cecs.pdx.edu/~timm/dm/cocomo.html>, que automatizam o COCOMOII. A **Figura 2** apresenta uma tela que ilustra os resultados fornecidos pelo software desenvolvido pela Universidade do Sul da Califórnia (University of Southern California – USC) e distribuído gratuitamente em conjunto com o livro que define o COCOMOII.

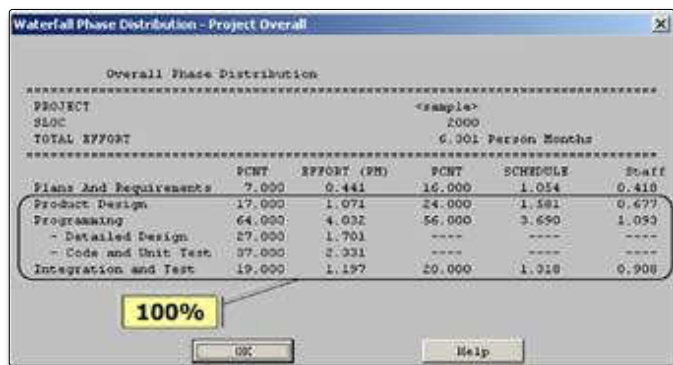


Figura 2. Distribuição geral de fases considerando a estratégia de desenvolvimento em cascata no que se refere ao esforço, prazo e equipe.

Observe na **Figura 2** que, quando os percentuais das colunas de esforço (PCNT relativos às colunas EFFORT e SCHEDULE) são totalizados, o resultado encontrado ultrapassa os 100%.

A fase de planejamento e requisitos representa 7% do total do esforço estimado pelo modelo. Quando o modelo fornece a estimativa de 0,441 Pessoas-Mês para essa fase, isso é feito por extrapolação (essa dinâmica será melhor explicada adiante na **Figura 4**). Tal situação acontece por que pesquisas da equipe do COCOMOII verificaram que essa fase apresenta uma variação muito grande entre os 181 projetos utilizados na sua elaboração, tanto para o esforço (02 a 15%) quanto para o prazo (02 a 30%). Portanto, antes de usar o modelo, ele deve ser ajustado para que o percentual usado para essa fase seja aquele que se adéqüe ao contexto em que será usado. Por exemplo, se em minha organização esse percentual representar 15%, esse é o percentual que deve ser usado.

Isso não é exclusividade do modelo em cascata, havendo também extrapolações naqueles modelos baseados no RUP, como ilustrado na **Figura 3**.

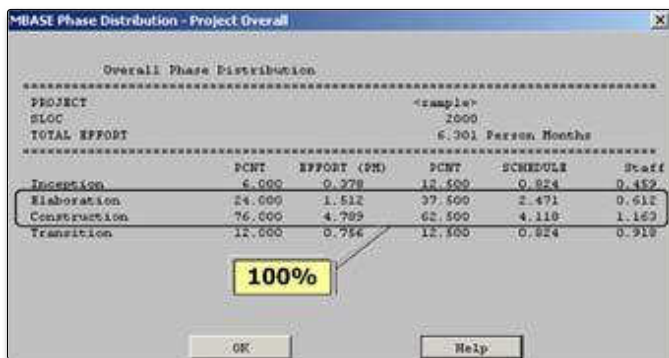


Figura 3. Distribuição geral de fases considerando a estratégia de desenvolvimento do RUP no que se refere ao esforço, prazo e equipe.

Por exemplo, considere que a aplicação do modelo em um projeto utilizando uma estratégia de desenvolvimento iterativa e incremental resulte em um esforço de 100 pessoas-mês, PM (ou 15.200 horas, já que o COCOMO utiliza a princípio 152 Hh / PM). O escritório de projetos verifica com base no levantamento dos projetos da mesma natureza que habitualmente a fase de Concepção representa 18% do esforço total do projeto e a fase de Transição, 20%. Portanto, restam para as outras fases 62% do total e essas 15.200 Hh correspondem a esse percentual. Por extrapolação, chega-se à conclusão que o projeto como um todo tenha um esforço estimado de 24.516 Hh, a fase de Concepção corresponda a 4.512 Hh e a de Transição 4.903 Hh. A **Figura 4** ilustra essa dinâmica da extrapolação aplicada às fases não cobertas pelo COCOMOII.

Os tipos de trabalho incluídos na estimativa do COCOMOII

Um modelo de custos fornece como produtos as estimativas de esforço com base nos parâmetros informados. O que está incluído no escopo das estimativas em termos do tipo de trabalho

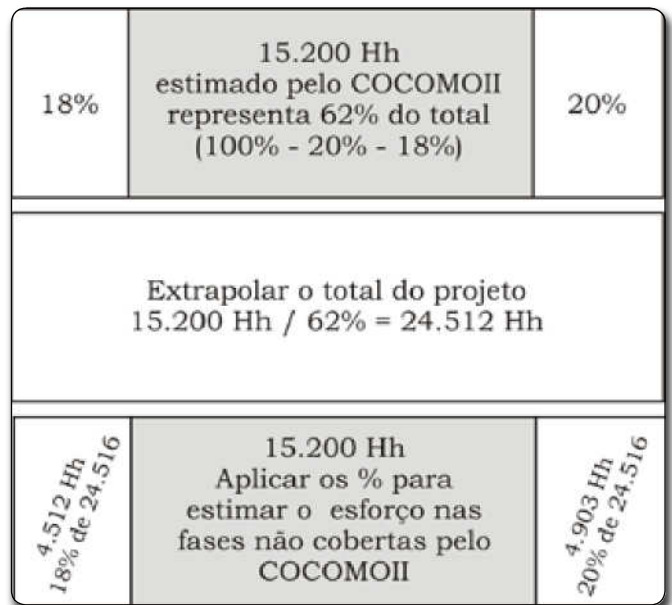


Figura 4. Dinâmica na extrapolação das estimativas para as fases não incluídas nas estimativas geradas pelo COCOMOII.

realizado? A resposta para o caso do COCOMOII é toda atividade cujo custo seja diretamente apropriado ao projeto.

Por exemplo, o trabalho de gerentes de projeto, gerentes de configuração, programadores, analistas de requisitos, arquitetos e analistas de teste podem facilmente ser alocados a um determinado projeto e o respectivo custo ser diretamente apropriado ao mesmo, e o esforço gerado pelo modelo ser representativo do mesmo.

Aquele trabalho considerado overhead, que não é facilmente alocado a um determinado projeto e, conseqüentemente, não pode ter o seu custo apropriado diretamente ao projeto não está presente em termos das estimativas de esforço do COCOMOII.

Esse trabalho está incluído nas estimativas de custo derivadas, na medida em que o custo médio da pessoa-mês computa a bonificação sobre despesas indiretas (BDI) que visa custear aquilo não ponderado pelo esforço estimado pelo modelo. Exemplos desse tipo de trabalho, que não estão incluídas nas estimativas de esforço do COCOMOII, é o departamento de pessoal, secretárias e executivos.

Tal qual a normalização das fases, a normalização das diferentes naturezas de trabalho tem um papel muito importante na estimativa e na análise do realizado.

A acuidade da estimativa conforme se avança

O COCOMOII traz as faixas de incerteza conforme a fase em que se está no ciclo de vida aos moldes do exposto na primeira parte do artigo da edição anterior - seção "Outras Formas de Representar e Lidar com a Incerteza".

A **Figura 1** também é conhecida como o "Cone da Incerteza" em função do formato que assume ao representar o aumento da acuidade das estimativas na medida em que os riscos de escopo e produtividade diminuem. A **Tabela 4** apresenta essas faixas de incerteza.

Fase	Estimativa Otimista	Estimativa Pessimista
Estudo de Viabilidade	0,25 x	4 x
Escopo Definido (LCR/IRR)	0,5 x	2 x
Planos e Requisitos Elaborados (SRR / LCO)	0,67 x	1,5 x
Projeto Preliminar Concluído (PDR / LCA)	0,8 x	1,25 x

Tabela 4. Níveis de incerteza nas diferentes fases do projeto.

Por exemplo, se uma estimativa for elaborada em um momento logo após o escopo estar definido (LCR) e, ao aplicar o modelo, obtenha-se um resultado de 15.200 Hh, deve-se considerar uma estimativa entre 7.600 Hh e 30.400 Hh. Já se isso acontece ao final da fase de planos e requisitos, deve-se considerar uma estimativa entre 10.184 Hh e 22.800 Hh.

Na seção “Relação entre Escopo e Esforço” do artigo da edição anterior, é apresentado o conceito de fator primário de custo, de tamanho, e de fatores secundários. No COCOMOII, os fatores secundários são aqueles denominados drives de esforço (effort drivers) e fatores de escala (scale factors) e buscam capturar as particularidades do objeto da estimativa quanto ao produto, ao projeto, à plataforma e à equipe.

Por exemplo, uma das particularidades a ser considerada em uma estimativa é o grau de compressão ou distensão do cronograma em relação ao prazo mais curto em que se utiliza o menor esforço (denominado “nominal” no vocabulário do COCOMOII). Nele, esse grau é ponderado pelo driver denominado SCED.

Dependendo do momento em que se esteja no ciclo de vida, da informação disponível e do segmento de mercado em que o objeto da estimativa estiver inserido, o COCOMOII oferece diferentes modelos, endereçando as diferentes necessidades e restrições desses contextos. Basicamente, o que distingue esses modelos é a quantidade de fatores secundários de custo que devem ser avaliados por quem está estimando. A Figura 5 apresenta a segmentação utilizada na definição do COCOMOII.



Figura 5. Segmentos de mercado no desenvolvimento e manutenção de software utilizado no COCOMOII.

Considerando que o esforço envolvido no segmento de programação pelo usuário final esteja na ordem de até 40 horas, ou menos, e o exposto no artigo da edição anterior na seção “A Ordem de Grandeza: Uma Quinta Dificuldade ao Estimar”, os

profissionais inseridos nesse segmento não necessitam de um modelo de estimativas como o COCOMOII.

Para os outros segmentos, o COCOMOII oferece três modelos: a) Composição de Aplicação (Application Composition) com três parâmetros; b) Projeto Preliminar (Early Design) com 17 parâmetros; e c) Pós-Arquitetura (Post-Architecture) com 24 parâmetros.

O segmento denominado Composição de Aplicação corresponde aos projetos utilizando ambientes de desenvolvimento integrado (IDE), normalmente utilizadas em projeto de médio/pequeno porte e que consistem basicamente de compor uma aplicação com base em componentes visuais. O COCOMOII propõe a utilização de simples relações entre tamanho e esforço como discutido na seção “Modelos baseados em Simples Relações Usando a Análise de Pontos de Função” do artigo da edição anterior.

Para os demais segmentos, o COCOMOII define dois modelos: O Modelo Projeto Preliminar (Early Design) e o Modelo Pós-Arquitetura (Post-Architecture). Não há obrigatoriedade em usar necessariamente um determinado modelo em determinada fase ou segmento. A Figura 6 ilustra isso conforme a fase em que se esteja e o nível de informação disponível.

A diferença entre os dois modelos pode ser resumida na quantidade de drivers de esforço utilizados na elaboração da estimativa que devem ser informados pelos seus usuários e a confiabilidade esperada. Tanto o modelo pós-arquitetura quanto o projeto preliminar apresentam confiabilidade de 80%, 10% acima da estimativa otimista e abaixo da pessimista.

Drivers de Esforço e Fatores de Escala x Valor do Fator de Ajuste

Os drivers de esforço e fatores de escala são definidos conforme o modelo e apresentam uma série de orientações para fins de enquadramento do caso em análise quanto a uma escala. Nesse particular, a dinâmica é bastante similar àquela estabelecida pelo IFPUG para determinação do nível de influência de cada característica geral de sistema (GSC) no processo de determinação do valor do fator de ajuste (VAF). Destaca-se, nesse aspecto, por as semelhanças cessarem nisso.

Ao contrário do modelo de VAF onde existe um piso e teto fixos, únicos para a avaliação do impacto de cada característica, no COCOMOII cada driver de esforço e cada fator de escala têm um piso e teto específicos. Por exemplo, um dos drivers do modelo é a confiabilidade exigida do software (reliability) e, quando no caso em análise houver indicação de risco à vida humana (condição de teto para o mesmo), estima-se haver um impacto de +26% ao esforço nominal estimado para sua entrega, já quando houver indicação de uma pequena inconveniência (condição de piso para o mesmo) o estima-se um impacto de -18%. Outro exemplo é o driver complexidade do produto (complexity) que pondera a maior ou menor complexidade em operações de controle, cálculos, operações dependentes de dispositivos, manipulação de dados e interface com usuário, o seu piso representa um impacto de -27% e, o seu teto, +74%.

Outra diferença é que o modelo do VAF é linear, enquanto no COCOMOII os fatores de escala têm um impacto exponencial no esforço nominal, em outras palavras, o modelo inclui economias e

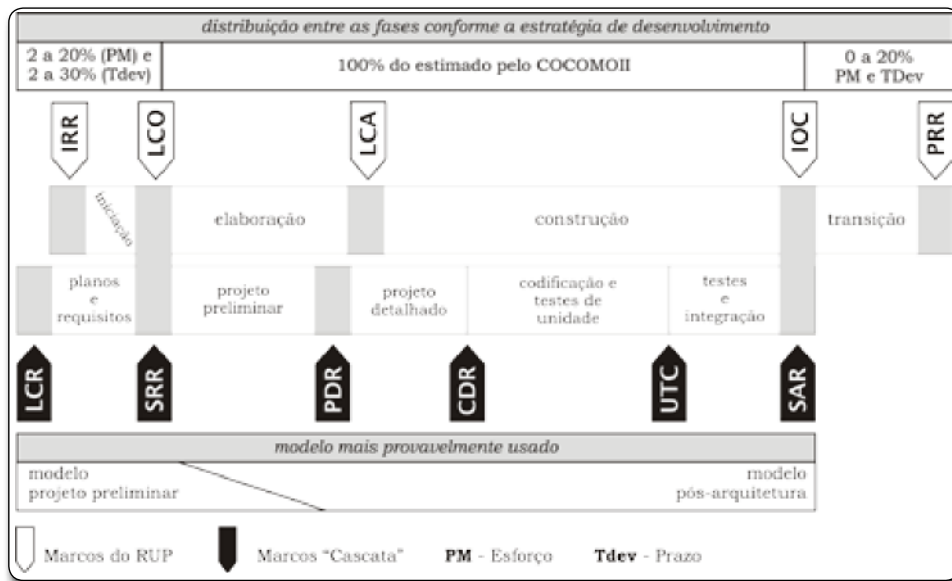


Figura 6. Momento mais provável em que determinado modelo seja o mais adequado.

deseconomias de escala. A seção “Relação entre escopo e esforço”, do artigo da edição anterior, explica o papel da consideração dos fatores com impacto exponencial em um projeto de software.

Uma das funções do modelo é permitir aos seus usuários, ao analisar um caso, estabelecerem um ponto entre o respectivo piso e teto em um processo interpretativo das orientações fornecidas (subjetivo), mas ao contrário da determinação do VAF, esses percentuais são aplicados ao esforço e não ao tamanho e, portanto, há a possibilidade de amortecer essas diferentes interpretações ao ajustar as constantes de produtividade do modelo (desde que haja uma uniformidade quanto a essas interpretações, ao menos no âmbito da organização em que os dados estão sendo coletados, consolidados e analisados).

Além das aplicações dos fatores de escala e drivers de esforço em estimativas usando o COCOMOII, o conceito pode ser usado como uma referência ou inspiração para a criação de critérios de similaridade utilizados para análise das métricas de software em busca da criação de categorias de produtividade, ou na busca de indicadores estatísticos que apoiem o processo de planejamento.

Por exemplo, os autores tiveram uma experiência em que foi sugerida a criação de três critérios em que uma demanda deveria ser avaliada para fins de registro na base histórica: (a) Conhecimento do Usuário Sobre o Negócio (apelidada de CUSN); (b) Conhecimento do Analista Sobre a Aplicação (apelidada de CASA); e (c) foi utilizado um dos drivers do COCOMOII, especificamente, um que mede a familiaridade com o problema (cuja sigla é PREC) e cujas orientações para classificação estão na Tabela 5 para fins de ilustração.

A fórmula do COCOMOII – o coração dos modelos

A Figura 7 apresenta a fórmula do COCOMOII. Dificilmente será necessário utilizá-la manualmente e está presente neste texto para fins didáticos e, o mais provável, é que o seu usuário utilize alguma ferramenta ou planilha para sua utilização.

Característica	Muito Baixa	Nominal-Alta	Altíssima
Entendimento organizacional dos objetivos do produto	Geral	Considerável	Completo
Experiência trabalhando com sistemas de software afins	Moderada	Considerável	Extensiva
Desenvolvimento concorrente de novos equipamentos e procedimentos operacionais	Extensivo	Moderado	Algum
Necessidade de inovar arquiteturas de processamento de dados e algoritmos	Considerável	Algum	Mínimo

Tabela 5. Exemplo das orientações fornecidas pelo COCOMOII para classificação dos fatores de custo secundário, especificamente do fator de escala familiaridade (Precedentness – PREC).

Nessa ferramenta, informará o tamanho do projeto e/ou de seus componentes e procederá a avaliação dos drivers de esforço e fatores de escala.

Apesar do estabelecido anteriormente, entender a fórmula ajuda a entender o funcionamento do modelo, suas entradas e seus produtos.

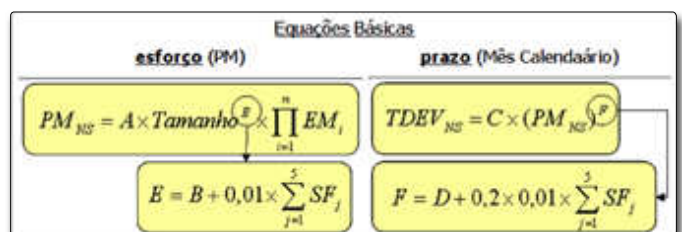


Figura 7. Fórmulas de esforço e prazo do COCOMOII.

Os produtos do modelo são representados pela variável TDEVNS, referindo-se ao prazo nominal expresso em meses calendário, e pela variável PMNS, referindo-se ao esforço nesse prazo nominal expresso em pessoas mês (vide explicação

na seção Relação entre Escopo e Esforço do artigo da edição anterior, para um melhor entendimento).

As constantes A, B, C e D representam a produtividade e o desempenho local, as duas primeiras no que se refere ao esforço e as duas últimas ao prazo. Antes de utilizar o modelo de estimativa, é necessário que essas constantes sejam calibradas às condições locais. Calibrar constantes de um modelo de estimativa é utilizar valores que sejam adequados ao contexto em que a estimativa será utilizada e aos objetivos que se deseja alcançar. A calibração é feita com base em experiências passadas compatíveis com esse contexto. O manual do COCOMOII recomenda que sejam utilizados pelo menos cinco casos para calibrar as constantes com efeitos multiplicativos A e C, e pelo menos dez projetos para calibrar as constantes com efeitos exponenciais B e D.

Os Drivers de Esforço

Na fórmula do COCOMOII (Figura 7), os drivers de esforço estão representados pela variável EM_i e os fatores de escala pela variável SF_i . No modelo Pós-arquitetura, serão dezessete os drivers de esforço (Figura 8) e no projeto preliminar, sete (Figura 9).



Figura 8. Drivers de esforço no modelo Pós-arquitetura; o driver de compressão de cronograma é único para todo o projeto e não é considerado na estimativa do esforço e prazos nominais.



Figura 9. Drivers de esforço no modelo Projeto Preliminar e a correspondência entre os drivers de esforço no modelo Pós-Arquitetura.

EM_i	Muito Baixo	Baixo	Nominal	Alto	Muito Alto	Altíssimo
RELY	0.82	0.92	1.00	1.10	1.26	-
DATA	-	0.90	1.00	1.14	1.28	-
CPLX	0.73	0.87	1.00	1.17	1.34	1.74
RUSE	-	0.95	1.00	1.07	1.15	1.24
DOCU	0.81	0.91	1.00	1.11	1.23	-
TIME	-	-	1.00	1.11	1.29	1.63
STOR	-	-	1.00	1.05	1.17	1.46
PVOL	-	0.87	1.00	1.15	1.30	-
ACAP	1.42	1.19	1.00	0.85	0.71	-
PCAP	1.34	1.15	1.00	0.88	0.76	-
PCON	1.29	1.12	1.00	0.90	0.81	-
APEX	1.22	1.10	1.00	0.88	0.81	-
PLEX	1.19	1.09	1.00	0.91	0.85	-
LTEX	1.20	1.09	1.00	0.91	0.84	-
TOOL	1.17	1.09	1.00	0.90	0.78	-
SITE	1.22	1.09	1.00	0.93	0.86	0.80
SCED	1.43	1.14	1.00	1.00	1.00	-

Tabela 6. Drivers de esforço do modelo Pós-arquitetura

A Tabela 6 apresenta os diferentes valores referentes aos drivers de esforço no modelo Pós-arquitetura. Esses pesos também não serão informados pelos usuários do modelo, estando mais provavelmente incluídos no software ou na planilha que automatiza o COCOMOII. Nela, estão os valores para os 17 drivers de esforço do modelo Pós-arquitetura. O driver de compressão de cronograma (SCED) é global para todas as partes do projeto, enquanto os demais são avaliados para cada parte isoladamente.

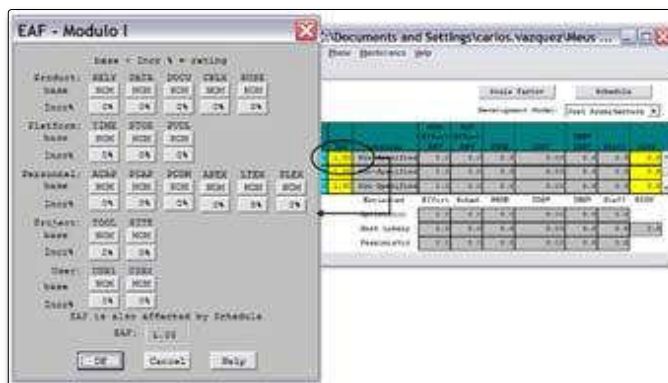


Figura 10. Formulário para informar a classificação dos drivers de esforço no modelo Pós-arquitetura ao estimar um projeto usando o COCOMOII.

A Figura 10 ilustra como é feito o registro da análise dos diferentes drivers também no modelo Pós-arquitetura.

Lidando com a compressão de cronograma – o driver SCED

Na estimativa de esforço nominal (PMNs), o driver de compressão de cronograma não é utilizado. O valor correspondente a sua avaliação na Tabela 6 deve ser incluído conjuntamente

com os valores referentes aos outros drivers no resultado da multiplicação quando se deseja considerar os seus efeitos na estimativa de esforço (que, com isso, deixa de ser nominal). A determinação do valor correspondente deve observar a Tabela 7, conforme o grau de compressão exigido.

Muito Baixo	Baixo	Nominal	Alto	Muito Alto	Altíssimo
75% do nominal	85%	100%	130%	160%	-

Tabela 7. Critérios para qualificação do driver de compressão de cronograma.

Para a estimativa do prazo, considerando essa compressão do cronograma, acrescenta-se mais um componente à respectiva fórmula do prazo nominal (TDEVns), como descrito na Fórmula 1.

$$TDEV = C \times (M_s)^F \times \frac{SCED\%}{100}$$

Fórmula 1. Fórmula de esforço considerando a compressão de cronograma.

Por exemplo, se em determinado projeto deseja-se considerar uma compressão de cronograma de 25% e, portanto, o prazo estimado será 75% do prazo nominal, o esforço considerando essa compressão deve ser 43% superior ao esforço nominal.

Os Fatores de Escala

Os fatores de escala são os mesmos, independentemente do modelo utilizado ser o Pós-arquitetura ou Projeto Preliminar, onde a Tabela 8 apresenta os valores para os cinco fatores de escala (SFi) e, a Figura 11, um exemplo de como eles são informados pelo usuário.

SF _i	Muito Baixa	Baixa	Nominal	Alta	Muito Alta	Impacto Máximo
PREC	6,20	4,96	3,72	2,48	1,24	
FLEX	5,07	4,05	3,04	2,03	1,01	1,26
RESL	7,07	5,65	4,24	2,83	1,41	1,39
TEAM	5,48	4,38	3,29	2,19	1,10	1,29
PMAT	7,80	6,24	4,68	3,12	1,56	1,43

Tabela 8. Valores para os cinco fatores de escala do COCOMOII.

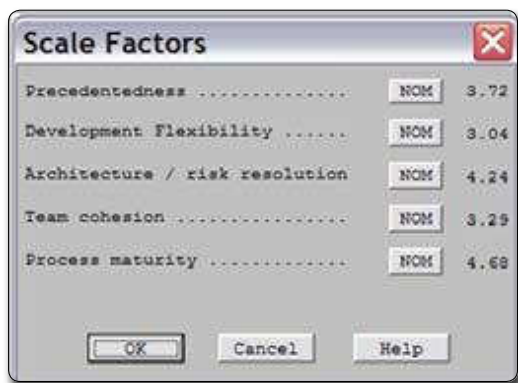


Figura 11. Formulário para informar a classificação dos fatores de escala ao estimar um projeto usando o COCOMOII.

O Fator Primário de Custo – O Tamanho

O tamanho é informado em milhares de linhas de código fonte (KSLOC). Essa unidade de tamanho é de difícil estimativa e a recomendação é que seja utilizada a estimativa em pontos de função para esse fim. Para isso, é necessário definir qual o fator de equivalência que será utilizado ao converter de pontos de função não ajustados para KSLOC.

O fator de equivalência não será constante, único, para qualquer aplicação ou projeto, mas para efeitos da estimativa, é um valor único necessário para converter de pontos de função para KSLOC. Usar a mediana de uma amostra é uma boa opção, já que a mediana é menos afetada que a média por valores extremos.

A Tabela 9 apresenta uma série de projetos onde a mediana do fator de equivalência é de 99,49 SLOC/PF. A mediana divide uma amostra ordenada em duas partes, onde uma parte contém a metade dos elementos da amostra que são menores ou iguais à mediana, enquanto a segunda metade contém os demais elementos, maiores ou iguais à mediana.

Projeto	PF	SLOC	SLOC/PF
Projeto A	234	19.209	82,09
Projeto B	560	75.252	134,38
Projeto C	351	29.537	84,15
Projeto D	430	53.964	125,50
Projeto E	103	24.045	233,45
Projeto F	263	26.167	99,49
Projeto G	2.024	188.900	93,33
Projeto H	1.623	144.054	88,76
Projeto I	850	94.305	110,95

Tabela 9. Insumos para determinar o fator de equivalência entre a quantidade de pontos de função estimados e o fator de custo primário como esperado pelo COCOMOII.

A esse processo é dado o nome de “backfiring” e, quando utilizado no contexto do COCOMOII, os ruídos advindos da introdução desse passo no processo de estimativa tendem a ser anulados pelo processo de calibração das constantes de produtividade e desempenho. O processo inverso de converter de KSLOC para pontos de função deve ser evitado na medida em que normalmente não há a amortização desses ruídos como no COCOMOII.

Portanto, será utilizado como tamanho na fórmula do COCOMOII o valor de 49,475 KSLOC referentes a um projeto cujo tamanho funcional seja estimado em 500 PF (500 PF x 99,49 SLOC/PF / 1000).

O COCOMOII também prevê a consideração de reuso e código gerado automaticamente que não serão tratados neste texto. A Figura 12 ilustra como as informações de tamanho são imputadas no COCOMOII com a aplicação desenvolvida pela Universidade da Califórnia do Sul, apresentada anteriormente. Nela há o formulário para informar o tamanho, e o campo REVL corresponde ao percentual de retrabalho advindo da volatilidade dos requisitos.

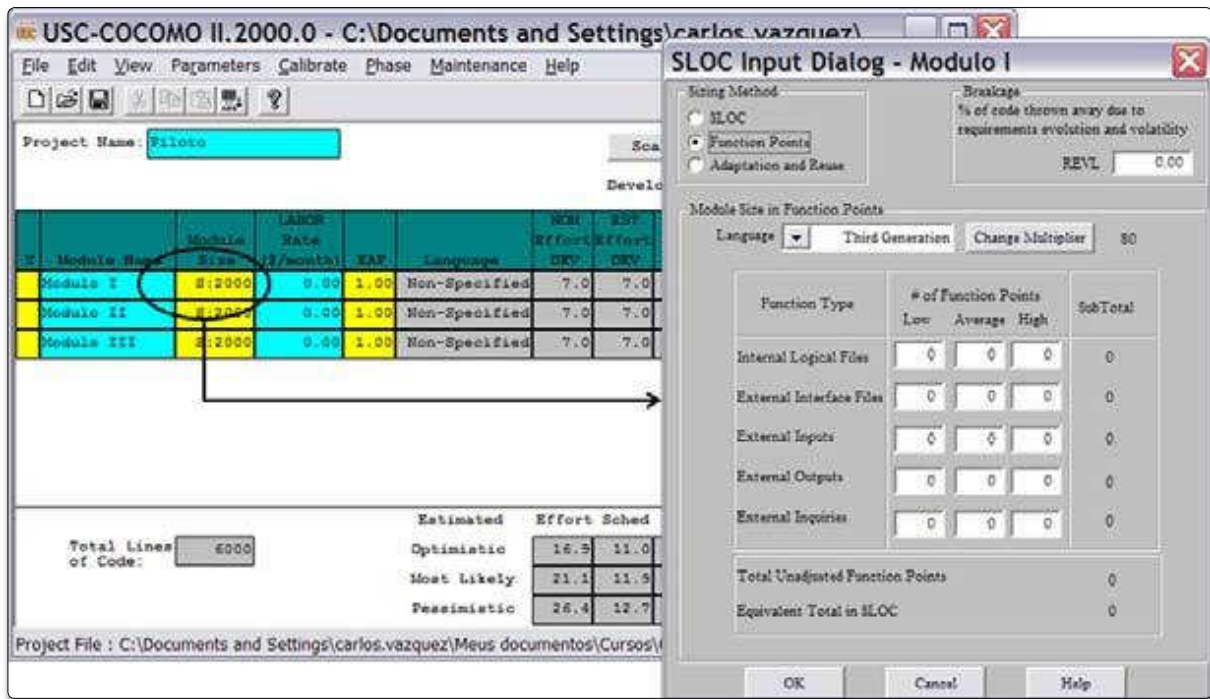


Figura 12. Entrada dos dados de tamanho – o fator primário de custos.

A Necessidade de Calibrar do Modelo

Este texto abordou até este ponto a utilização do modelo COCOMOII a partir de parâmetros de tamanho e da análise qualitativa de uma série de aspectos relacionados ao projeto, produto, usuário e processo. A partir dos mesmos, o modelo produz estimativas de esforço, prazo e tamanho da equipe (razão entre os dois itens anteriores). Mas não se deve utilizar o modelo simplesmente a partir disso!

No modelo de estimativa discutido no artigo da edição anterior, referente ao deslocamento entre o Rio de Janeiro e Niterói, considerou-se uma velocidade média de 25 Km/h. Nesse artigo foi definido um modelo de estimativa que pode ser descrito como “A estimativa de tempo para deslocar-se de um ponto A até um ponto B é a razão da distância e a velocidade média de 25 Km/h”.

Esse modelo está adequado para o ponto A ser o Centro do Rio de Janeiro e o ponto B, o Centro de Niterói, em um deslocamento feito utilizando um automóvel. Está adequado se o ponto A for a Barra Funda em São Paulo e o Ponto B o Morumbi? Talvez outra velocidade média seja mais representativa que não os 25 Km/h utilizados no modelo. Descobrir essa nova velocidade média é calibrar o modelo.

Calibrar o modelo envolve definir um objetivo a ser alcançado como, por exemplo, diminuir o erro médio. Para tanto, são necessárias fotografias de eventos passados e que sejam utilizados na otimização buscando esse objetivo. A ferramenta que utilizamos até o momento para ilustrar o COCOMOII permite a calibração de suas constantes de esforço a partir da informação do realizado pelo usuário. A Figura 13 ilustra como isso é feito e apresenta o formulário para informar o esforço e o prazo realizados de um projeto onde os parâmetros do modelo foram analisados e

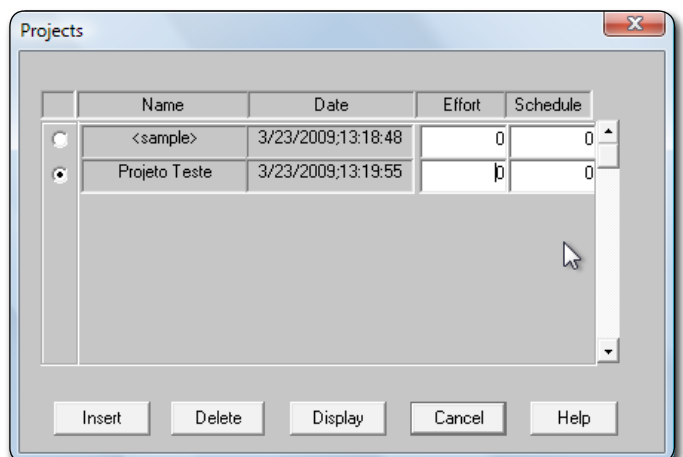


Figura 13. Informação do esforço e prazo realizados para calibrar o modelo.

informados (Tamanho, Drives de Esforço, Fatores de Escala, entre outros).

Além do software utilizado na Figura 13, outros também gratuitos estão disponíveis como o CALICO da SoftStar Systems. O uso de uma ferramenta para esse fim não é um requisito, desde que haja entendimento sobre as diferentes estratégias de medir o erro de modelos de estimativa.

Com uma planilha eletrônica é possível a realização de ensaios onde diferentes critérios de avaliação são utilizados, e o responsável pelo trabalho pode escolher as constantes que mais se adequem às suas necessidades. Ao proceder a uma calibração manual é fundamental normalizar o esforço e prazo de tal forma que sejam considerados os seus valores nominais, desconsiderando os efeitos da compressão de cronograma e que todos os prazos expressos em pessoa-mês sejam referentes a um mesmo número de homens-hora por mês-calendário.

Discutir sobre a calibração de modelos de estimativa como o

COCOMOII envolve apresentar os diferentes meios de medir o erro como, por exemplo: (a) a Média da Magnitude do Erro Relativo - estimada, realizada e balanceada; (b) o grau de predição do modelo; (c) a soma dos quadrados dos erros. O espaço disponível para este artigo não permite fazer isso, que será tratado em uma nova oportunidade.

Considerações Finais

Muitos vêem o COCOMOII como um software da Universidade do Sul da Califórnia. Com base no exposto neste texto, percebe-se que ele é muito mais do que isso e, pelo contrário, o software em questão é apenas uma das implementações do modelo, e das mais rudimentares. Outros pacotes comerciais mais completos e com dados de calibração mais abrangentes estão disponíveis como o Cost Xpert e o CoStar, contudo o principal valor está na rigorosa definição de premissas e em sua simplicidade. ●

Referências

Boehm, B. et al., "Software Cost Estimation With COCOMO II", Prentice Hall, 2000.

Aguiar, M., "COCOMOII Local Calibration Using Function Points", <http://sunset.usc.edu/events/2005/COCOMO/presentations/CIIILocalCalibrationPaper.pdf>

Ligett, D., "Techniques for Calibrating COCOMO Estimating Models", <http://www.softstarsystems.com/calico.htm>.

Maxwell, K. D., "Applied Statistics for Software Managers", Prentice Hall, 2002.

Dê seu feedback sobre esta edição!

A Engenharia de Software Magazine tem que ser feita ao seu gosto. Para isso, precisamos saber o que você, leitor, acha da revista! Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/esmag/feedback



Cursos Online



A Revista **.net Magazine** oferece para seus assinantes uma série de Cursos Online de alto padrão de qualidade .

Conheça abaixo os cursos já disponíveis:
www.devmedia.com.br/curso/netmagazine

- Crie uma loja Virtual completa
- Construindo relatórios com Crystal Reports e Visual Studio 2005
- Criando uma aplicação Web Completa
- Criando uma aplicação client/server no Visual Studio 2005
- Aprenda a criar um blog com ASP.NET
- Curso de C# * Curso em andamento

A sua melhor opção de aprendizagem!

Assine a **.net Magazine** e Comece já seu treinamento!
www.devmedia.com.br/assine