Estimativas Confiáveis de Prazos Para Gerentes de Projetos

POR MAURICIO AGUIAR

Você foi nomeado gerente daquele projeto importantíssimo, que vai alavancar os negócios da empresa nos próximos anos, a verdadeira menina dos olhos da diretoria. Um exército de consultores foi contratado, envolvendo as empresas de consultoria habituais nesse porte de projeto. Estão concluindo a elaboração de um anteprojeto preliminar, sendo a preocupação de todos a mesma de sempre: qual prazo será fornecido pelos gênios que cobram US$$$ por hora? E qual o custo estimado?

Todos sabem para onde os olhos se voltarão quando esses dados forem fornecidos: você. O prazo é tecnicamente viável? Não poderia ser mais curto? O tamanho da equipe é adequado? O custo é compatível com o porte do sistema?

Não há como dizer que mais um consultor será necessário para dar essas respostas. É o momento de assumir a responsabilidade. E seria bom que você soubesse exatamente o que está assumindo... Quais são as suas chances de sucesso?

Neste artigo, vamos ver algumas maneiras simples de suavizar o problema acima, dentro do espaço e tempo disponíveis. Isso porque produzir estimativas confiáveis para o desenvolvimento de sistemas é matéria complexa, que enche dezenas de gordos volumes escritos por persistentes "garimpadores" de dados, tais como Capers Jones, Grady, Roetzheim e outros. Não é o nosso caso. Aqui, vamos abordar o que os autores chamam sanity checks (literalmente, "cheques de sanidade"), ou seja, estimativas pouco precisas, mas que podem ser geradas a partir de pouca informação, servindo para validar estimativas elaboradas por outras pessoas, ou para que se tenha uma idéia inicial do tamanho, prazo e custo de um projeto.

A primeira coisa que precisa ser avaliada é o tamanho do projeto. Em se tratando de desenvolvimento de sistemas, como é o nosso caso, o tamanho é freqüentemente denominado volume. Este pode ser medido em linhas de código, pontos de função ou em outras unidades menos utilizadas. Para sistemas cuja dimensão principal é o dado, que têm muitas entradas e saídas e não contêm algoritmos complexos, o ponto de função é a medida indicada. Essas características são comuns à maioria dos sistemas comerciais, por isso usaremos pontos de função (PF) em nossos exemplos.

Estimando o volume em pontos de função

Para que o leitor ganhe alguma sensibilidade com o PF como medida de volume, mostramos o tamanho aproximado de algumas aplicações típicas, medido em pontos de função [1]: ferramenta CASE IEF (Texas) – 20.000; compilador MS Visual Basic – 3.000; SGBD IMS (IBM) – 3.500; gerenciador de TP CICS (IBM) – 2.000; MS Word 7.0 – 2.500; MS Excel 6.0 – 2.500; MS Project – 3.000; imposto de renda pessoal – 2.000; contabilidade geral – 1.500; processamento de pedidos – 1.250; recursos humanos – 1.200; suporte a vendas – 975; preparação de orçamento – 750.

Tendo obtido uma idéia das ordens de grandeza envolvidas, vejamos a primeira técnica para estimativa de volume em PF.

Estimando o volume a partir da natureza da aplicação

Capers Jones [1] desenvolveu um modelo para a estimativa do volume de um sistema a partir da natureza da aplicação. Basta consultar três tabelas e... pronto! Vejamos como fazer isso.

Primeiro, classificamos a aplicação segundo o escopo (Tabela 1). Seja, por exemplo, um "novo sistema". Guardamos o índice de escopo = 9.

Em segundo lugar, classificamos a aplicação conforme a classe (Tabela 2). Em nosso caso, imaginamos que o sistema vai rodar em nossa empresa, em vários sites. Guardamos o índice de classe = 5.

Em terceiro lugar, classificamos a aplicação segundo o tipo (Tabela 3). Seja uma aplicação cliente/servidor. Teremos, então, o índice de tipo = 8. Somando os três índices obtidos, teremos: 9 + 5 + 8 = 22. A estimativa de Capers Jones é obtida elevando-se o resultado da soma a 2,35. Em nosso caso, chegamos a 1.428 PF. Nosso sistema ficou, então, na faixa de 1.000-1.500 PF, o que é consistente com os dados anteriormente apresentados.

Tabela 1

Índice	Escopo	Índice	Escopo
1	Sub-rotina	6	Programa isolado
2	Módulo	7	Componente de sistema
3	Módulo reutilizável	8	Versão de sistema
4	Protótipo descartável	9	Novo sistema
5	Protótipo evolutivo	10	Sistema composto

ESTIMANDO O VOLUME A PARTIR DO MODELO DE DADOS E DAS INTERFACES

Nesse método, vamos supor que é possível gerar um modelo de dados "tentativo" para a aplicação a ser desenvolvida. Com base nessas informações, obteremos uma estimativa para a quantidade total de PF do sistema [2]: PF = Qtd. (Arquivos ou Tabelas) X 35. Embora essa medida seja apenas uma aproximação inicial, ela pode ser melhorada, se utilizarmos mais algumas informações.

Considere apenas entidades "fortes" na contagem, isto é, aquelas cuja existência não dependa de outras. Em bancos de dados relacionais, por exemplo, é comum a criação das tabelas Nota Fiscal e Item NF, para representar uma única nota fiscal do mundo real. Nesse caso, uma linha da tabela Item NF não pode existir sem a correspondente Nota Fiscal. Por essa razão, contaríamos apenas uma tabela.

Outra maneira de explicar isso é dizer que, na Análise de Pontos de Função, os elementos contados correspondem à visão do usuário. Este certamente vê o conjunto Nota Fiscal + Item NF como um único arquivo lógico – Nota Fiscal.

Caso o sistema consulte tabelas ou arquivos de outros sistemas, conte-os também. Esses arquivos não devem ser atualizados, mas apenas consultados pelo sistema a ser estimado. Por isso, têm um peso menor na contagem, devendo sua quantidade ser multiplicada por 15. A fórmula completa fica, então: PF = (Qtd. Arquivos ou Tabelas X 35) + (Qtd. Interfaces X 15).

Essa forma de contagem de PF, criada pela Associação de Métricas da Holanda, é denominada *indicativa*. Vejamos como fica essa medida de volume para algumas quantidades de tabelas, sem considerar as interfaces externas (Tabela 4).

Parece razoável supor que o sistema exemplificado, com cerca de 1.400 PF, venha a ter mais ou menos 40 tabelas independentes, conforme indicam os dados.

ESTIMANDO PRAZOS E RECURSOS A PARTIR DO VOLUME

Uma vez conhecido o volume da aplicação em pontos de função, resta estimar o prazo ótimo de desenvolvimento e o tamanho ideal da equipe. É importante notar que não existe um único prazo de desenvolvimento, dado um certo volume. O prazo vai depender do custo incorrido, o qual reflete a quantidade de recursos alocados. A relação entre essas variáveis pode ser ilustrada pela figura constante no site da *DevMag*.

Td é o tempo considerado ótimo para o desenvolvimento, porque:

- De Td para a esquerda, os custos aumentam desproporcionalmente em relação ao tempo ganho.
- De Td para a direita, o tempo aumenta desproporcionalmente em relação à economia alcançada.

To é o tempo de desenvolvimento que acarreta o menor custo. A relação entre Td e To é: To = 2 Td.

A região em amarelo é denominada Região Impossível porque, em pesquisas realizadas com mais de 750 projetos [3], nenhum deles foi concluído satisfatoriamente em um prazo menor ou igual a esse. Essa região começa a 75% de Td, isto é, se Td for igual a 10 meses, será impossível executar o projeto em 7,5 meses ou menos.

ESTIMANDO O PRAZO E O ESFORÇO A PARTIR DA APROXIMAÇÃO DE CAPERS JONES

O cálculo rigoroso do tempo ótimo Td acima definido exige ferramentas de software sofisticadas. Inicialmente, utilizaremos a aproximação de Capers Jones [1]: Td (em meses) = V ** t, onde o expoente t varia de 0,32 a 0,45, dependendo do ambiente. Os seguintes valores são típicos: sistema comum – 0,32-0,35; sistema orientado a objetos – 0,36; sistema cliente/servidor – 0,37; sistema terceirizado – 0,38; sistema de informações gerenciais – 0,39; programa produto comercial – 0,40; programa de sistema operacional – 0,41; software militar – 0,43-0,45.

O sistema cliente/servidor anteriormente exemplificado poderia ter seu prazo ótimo definido por Td = 1.428 ** 0,37 = 14,7 meses. Quanto ao tamanho ótimo da equipe, Capers Jones considera que, na prática, cada integrante gera cerca de 150 pontos de função ao longo de um projeto. Dessa forma, o tamanho da equipe seria obtido através de: Te = FP / 150. Para o nosso exemplo, Te = 1.428 / 150 = 9,5 pessoas. O esforço resultante seria, então, E = 9,5 X 14,7 = 139,7 homens-meses.

Como a produção de cada pessoa independe do volume, fica claro que a fórmula acima é meramente uma aproximação inicial. Apesar de ser um ponto de partida, a aproximação de Capers Jones não leva em conta muitos fatores, especialmente a linguagem adotada no desenvolvimento. Vejamos, a seguir, um método um pouco mais preciso.

Tabela 2

Índice	Classe	Índice	Classe
1	Software individual	9	Internet
2	Shareware	10	Software alugado
3	Software acadêmico	11	Software bundled
4	1 Site - Interno	12	Software comercial (*)
5	Multi-site - Interno	13	Contrato de outsourcing
6	Projeto contratado - Civil	14	Contrato governamental
7	Time sharing	15	Contrato militar
8	Serviço militar		

(*) – programas como Word, Excel, etc.

Tabela 3

Índice	Tipo	Índice	Tipo
1	Não-procedural	11	Comunicações
2	Web applet	12	Controle de processo
3	Batch	13	Sistema confiável (trusted)
4	Interativa	14	Sistema embedded
5	GUI interativa	15	Processamento de imagem
6	Batch DB	16	Multimídia
7	DB interativa	17	Robótica
8	Cliente/servidor	18	Inteligência artificial
9	Matemática	19	Rede neural
10	Sistema operacional	20	Híbrido: Misto

ESTIMANDO O PRAZO E O ESFORÇO A PARTIR DE UMA FERRAMENTA

Há uma boa quantidade de ferramentas para estimativas, quase todas baseadas no modelo COCOMO (COnstructive COst MOdel), apresentado por Barry Boehm em 1981 [5] e agora em sua segunda versão (COCOMO II). A seguir, apresentamos uma versão simplificada do procedimento contido nessas ferramentas, de modo a possibilitar sua utilização manual.

A quantidade de comandos necessária para gerar um ponto de função varia muito, de acordo com a linguagem de programação utilizada. A seguir, indicamos a quantidade de comandos equivalente a um PF para cada linguagem [3,4]: Assembler – 320; Clipper DB – 40; Easytrieve + – 13; C – 128; Access – 38; IEF – 14; COBOL II – 107; Visual C++ – 34; MUMPS – 19; ADS/O – 20; Delphi 4 – 22; VB 5 – 29.

A ferramenta Cost Xpert [3] utiliza as seguintes fórmulas para calcular o Esforço (E) e o Tempo Ótimo (T) para um dado volume. Este é medido em milhares de linhas de código-fonte (KLOC) E = 2,4 X (V ** 1,05), sendo V = as milhares de linhas (sem os comentários) e E = homens-meses.

A quantidade de linhas de código pode ser obtida do volume em PFs, usando-se os dados do segundo parágrafo deste item. Para nossa aplicação, os 1.428 PF serão equivalentes a 1.428 X 29 = 41.412 comandos de Visual Basic 5.0 (VB5). O esforço será, então, E = 2,4 X (41,41 ** 1,05) = 119,7 homens-meses. A fórmula para o tempo ótimo de desenvolvimento é T = 2,5 X (E ** 0,32), sendo que T = meses e E = homens-meses, calculado pela fórmula anterior.

No nosso caso, teremos T = 2,5 X (119,7 ** 0,32) = 11,6 meses. Note-se que o valor encontrado anteriormente foi 14,7 meses, o que mostra a pouca precisão de tais fórmulas. Ainda assim, já podemos supor que tal sistema dificilmente ficará pronto em um prazo menor do que 1 ano. Dividindo-se 119,7 por 1,6, chegamos à alocação média de 10,3 pessoas, o que nos dá uma idéia do tamanho da equipe.

Caso resolvêssemos programar nossa aplicação em C++, os valores seriam outros. O volume em linhas de código seria 1.428 X 34 = 48.552 comandos, ou 48,55 KLOC. O Esforço (E), então, seria igual a 2,4 X (48,55 ** 1,05) = 141,5 homens-meses. O tempo ótimo seria 2,5 X (141,5 ** 0,32) = 12,2 meses. O tamanho médio da equipe seria 141,5/12,2 = 11,6 pessoas. O aumento de prazo e recursos deve-se à diferença de produtividade entre as duas linguagens. De acordo com o segundo parágrafo, VB5 é 34/29 = 1,17 vezes mais produtivo do que Visual C++.

CONCLUSÃO

Embora esses procedimentos simplificados não sejam precisos para sua utilização em orçamentos, fornecem uma idéia inicial dos volumes, prazos e custos. Funcionam como sanity checks, conforme mencionamos. Para quem quiser praticar a arriscada Arte da Estimativa, é conveniente conhecer bem a Análise de Pontos de Função e ter uma boa idéia sobre como funcionam os modelos paramétricos, tais como o COCOMO II. Além disso, é praticamente mandatória a utilização de alguma ferramenta de software.

Em uma próxima oportunidade, veremos como gerar estimativas para partes selecionadas do projeto, por exemplo: quanto tempo vai levar a fase de Levantamento de Requisitos?

Visite o site do Function Point Users Group in Rio, em www.fpug-rio.com.br, para mais informações sobre métricas, estimativas e pontos de função.

REFERÊNCIAS

- [1] JONES, Capers. *Estimating Software Costs*. McGraw-Hill, 1998. (Consulte também www.spr.com, site da empresa do autor.)
- [2] NESMA (Netherlands Metrics Association). *Early Function Point Counting*. http://www.nesma.nl/english/earlyfpa.htm – INTERNET,1999.
- [3] ROETZHEIM, W. H. & BEASLEY, R. A. *Software Project Cost & Schedule Estimating – Best Practices*. Prentice-Hall, 1997. (Consulte também a *newsletter* gratuita em www. marotz.com, site da empresa do autor.)
- [4] *Software Productivity Research – Programming Languages Table*. http://www.spr.com/library/0langtbl.htm – INTERNET, 1999.
- [5] BOEHM, Barry. *Software Engineering Economics*. Prentice-Hall, 1981. (Consulte o site de Engenharia de Software da USC, onde trabalha o autor: http://sunset.usc.edu.)

■ *Mauricio Aguiar (mauricioaguiar@yahoo.com) é engenheiro, analista de sistemas e presidente do Function Point Users Group in Rio – www.fpug-rio.com.br.*

Tabela 4

Qtd. Tabelas	PF Indicativos	Qtd. Tabelas	PF Indicativos
10	350	60	2.100
20	700	70	2.450
30	1.050	80	2.800
40	1.400	90	3.150
50	1.750	100	3.500