Estimativas para Software

João Carlos Testi Ferreira*

03-10-2016

Resumo

Estimativas estão presentes no nosso dia-a-dia. Este texto apresentará aspectos de estimativa para software, apresentando o principal modelo usado no mercado, COCOMO II, e mostrando a importância de boas medidas para se obter boas estimativas.

Introdução

Estimativas são produzidas para planejamento e avaliação de atingimento de objetivos. Ao fazer estimativas tentamos antever situações que permitam que tomemos decisões acertadas com relação ao uso de recursos ou correções de execução. Podemos avaliar melhores soluções para um problema que está ocorrendo ou mesmo evitar que um problema aconteça.

Estimativas podem nos fornecer meios de saber em que parte do processo são necessárias ações, inclusão de mais recursos ou mesmo mudança na forma de trabalho. Quanto mais precisas e bem definidas forem nossas estimativas melhores poderão ser as decisões tomadas com base nelas.

Este texto apresenta uma visão geral de estimativas, a razão de medir e como estas estimativas são estabelecidas. Além deste texto, outros estão disponíveis para apresentar este tema que é de grande importância no uso da Engenharia de Software, seja no planejamento da execução como no controle de qualidade.

1 Porque medir?

Na apresentação da técnica de Análise de Pontos de Função, Carlos Eduardo Vazquez and Guilherme Siqueira Simões and Renato Machado Albert (2007), p. 17, responde o que motiva a medição é o problema de os requisitos serem descobertos e evoluírem durante o processo de desenvolvimento, e a distribuição de recursos precisa ser avaliada de forma adequada com relação a essas mudanças. Em um contexto de Gerência de Projetos, as medidas irão servir para o planejamento das atividades e distribuição de recursos (tempo e pessoas).

*joao.c.ferreira@edu.sc.senai.br

A questão de porque medir precisa ser mais bem entendida. O entendimento inadequado desta questão faz com que ou deixemos de medir algo que agregaria valor ou medirmos apenas por medir, sem agregar valor ao processo.

A medição funcional, como já discutido, possui como grande benefício a capacidade de medir de forma independente à tecnologia, permitindo obter medidas em momentos iniciais e que são possíveis de comparar de forma segura. Não há como saber, por exemplo, quantas linhas de código um programa irá ter até ser escrito e a quantidade de linhas dependerá muito do estilo de escrita do programador, da linguagem utilizada e das características do projeto.

O que pretendemos com a medida estabelece os momentos em que devemos medir, e o que queremos medir. Se o objetivo é ter o tamanho da aplicação para manter tamanhos de equipe de manutenção, a medida deve ser realizada a cada entrega do software, para se ter o tamanho total da aplicação (tamanho final). Por outro lado, se o que se quer saber é a quantidade de mudanças que uma aplicação sofreu durante o processo de desenvolvimento, o que se deve medir são os pedaços de software que mudaram durante esse processo. Ou seja, o que irá determinar o que medir e quando medir é a intenção com a qual medimos.

1.1 Balanced Scorecard

Para poder medir o desempenho de empresas, Kaplan e Norton criaram um método chamado Balanced Scorecard. No entanto, conforme Costa (2006), p. 4, a sua forma de implementação apresenta muito potencial para permitir o alinhamento estratégico da organização.

O que se extrai de um modelo como o BSC é que a escolha de bons indicadores orienta a equipe para a direção que a empresa deseja seguir. Assim, esses indicadores norteiam a direção desejada e devem ser confiáveis o suficiente para ser a referência do atingimento de objetivos. Para isso o indicador deve ser baseado em boas medidas, colhidas de forma segura e no momento certo.

1.2 Custo e segurança da informação

Uma informação para controle deve ser o mais automática possível. Deve ser algo retirado do próprio processo, preferencialmente não dependendo de alimentação manual. Para que ela possa ser usada, deve ser comparada com uma medida confiável de tamanho, para que sua grandeza possa ser avaliada. Por exemplo, o total de erros identificados em uma aplicação foi de 100. Este número solto não diz muita coisa. Se estamos falando de uma aplicação de 50 PF, 100 pode ser muito, mas se estamos falando de uma aplicação de 5000 PF não é uma quantidade de erros significativa. Assim, para estabelecer o indicador usamos medidas (informações) obtidas do processo comparadas ao tamanho do que se mede.

2 Estimativa

Conforme Sommerville (2011), p. 442, existem duas técnicas para estimar: baseadas em experiência e modelagem algorítmica de custos. A baseada em experiência depende da experiência do gerente de projeto com projetos anteriores. A grande dificuldade desta forma de estimar é que projetos são únicos, com poucas informações que possam ser comparadas. Na modelagem algorítmica podem ser levados em consideração atributos

como característica do processo, produto e até experiência da equipe. De qualquer forma, a incerteza da estimativa é maior no início do projeto, e vai melhorando na medida que obtemos mais informações do processo.

Uma fórmula simples apresentada por Sommerville (2011), p. 443, para estimar esforço de projeto é:

$$Esforço = A \times Tamanho^B \times M$$

Na fórmula o **A** é constante e refere-se a práticas organizacionais e tipo de software. O **Tamanho** é uma medida de tamanho do software. O expoente **B** é algo entre 1 e 1,5, relacionado com a complexidade do sistema, e **M** é um multiplicador obtido pela combinação de atributos do processo, produto e desenvolvimento.

Estabelecer estes coeficientes e expoentes é algo complexo e que exige frequente calibração. Há uma carga de subjetividade significativa, o que torna a estimativa imprecisa.

2.1 COCOMO II

Um modelo popular de estimativa é o COCOMO II. Conforme Sommerville (2011), p. 444, ele foi desenvolvido à partir do modelo COCOMO (Constructive Cost Model).

Neste modelo fazem parte um submodelo de *composição de aplicação*, que permite estimar com base em reúdo de componentes, submodelo de *projeto preliminar*, que auxilia na estimativa de esforço inicial, submodelo de *reúso* que auxilia no cálculo de esforço para integração e um submodelo de *pós-arquitetura*, para uma estimativa depois que a arquitetura foi definida.

O capítulo 23 de Sommerville (2011), em seu item 5, apresenta técnicas de estimativa, entre elas o COCOMO II com algum detalhamento.

Há ferramentas na WEB que permitem cálculos para estimativas. A principal está no sítio da University of Southern California, no endereço:
http : *//csse.usc.edu/csse/research/COCOMOII/cocomo_main.html*.

3 Conclusão

Para obtermos boas estimativas precisamos ter boas medidas. O fundamento da estimativa é a medida. A estimativa, no entanto, só terá valor se usada para um objetivo bem definido. Toda estimativa possui um risco de erro que pode ser maior ou menor, conforme a fase do projeto em que estejamos e o volume de informação disponível que usamos para estimar. No entanto, mesmo com o erro esperado, a estimativa é valiosa, pois nos antecipa situações que podem fazer como uqe evitemos problemas ou mesmo nos fornece informações para identificarmos causas de problemas que estão ocorrendo no processo.

Referências

Carlos Eduardo Vazquez and Guilherme Siqueira Simões and Renato Machado Albert. *Análise de Pontos de Função*: Medição, Estimativas e Gerenciamento de Projetos de Software. 6. ed. São Paulo: Editora Érica, 2007. ISBN 9788571948990. Citado na página 1.

COSTA, A. P. *Balanced Socre Card*: conceitos e guia de implementação. São Paulo: Atlas, 2006. Citado na página 2.

SOMMERVILLE, I. *Engenharia de Software*: Uma abordagem profissional. 9. ed. São Paulo: Pearson Prentice Hall, 2011. ISBN 9788579361081. Citado 2 vezes nas páginas 2 e 3.