ENGENHARIA DE SOFTWARE APLICADA A COMPETIÇÕES DE PROGRAMAÇÃO: UMA PROVA DE CONCEITO[1]

MARCONDES, Francisco Supino[2];
INOCÊNCIO, Thiago de Jesus[3];

RESUMO

Este estudo procura avaliar a possibilidade do uso de técnicas de engenharia de software nas competições de programação. Como prova de conceito será utilizado o processo proposto pelo PSP no qual se procura prever o tempo e o tamanho das soluções de software por meio de métodos estatísticos. Após uma coleta inicial de dados, aplicou-se o processo proposto pelo PSP para estimar outros problemas semelhantes que na média teve 86.48% de precisão para as estimativas de linhas de código e 95.23% para as estimativas de tempo de desenvolvimento. Isso sugere que o PSP, em especial o método PROBE pode ser considerado para compor o ferramental utilizado em competições de programação ao menos para problemas *ad hoc*.
Palavras-chave: engenharia de software, competição de programação, processo de software.

ABSTRACT

This study aims to evaluate the possibility of using software engineering techniques in programming contests. As proof of concept it will be used the process proposed by the PSP that predict the time and the size of the software solutions by means of statistical methods. After an initial collection of historical data of solved problems, the process proposed by the PSP was applied to estimate other similar problems that in average had 86.48% of accuracy for the estimates of lines of code and 95.23% for the estimates of development time. This suggests that the PSP, especially the PROBE method can be used to compose the toolkit used in programming contests, at least for ad hoc problems.
KeyWords: software engineering, programming contests, software process.

INTRODUÇÃO

Competições de programação são eventos onde se apresentam problemas típicos de programação na forma de "desafios" com a intenção de que os concorrentes, em geral estudantes do ensino técnico ou graduação, os resolvam o mais rápido possível [1]. Dado que o tempo de desenvolvimento é uma variável crítica durante esse tipo de competição, este estudo se propõe a avaliar se a utilização de métodos estatísticos na previsão do tempo necessário para o desenvolvimento dos programas durante uma competição pode ser um diferencial competitivo. Este estudo é uma prova de conceito onde o método estatístico utilizado é o proposto pelo *Personal Software Process* (PSP) *cf.* [2] aplicado à problemas do tipo *ad hoc* (problemas de interpretação e adequação à regras [1]).

FUNDAMENTAÇÃO TEÓRICA

Os processos de software, de modo geral, são incentivados durante competições

[1]Projeto de iniciação científica voluntária.
[2]Mestre em Ciências da Computação; IFSP; São Paulo; São Paulo; supino@ifsp.edu.br; MARCONDES, Francisco
[3]Graduando em Análise e Desenvolvimento de Sistemas; IFSP; São Paulo; São Paulo; inocencio.thiago@hotmail.com; **INOCÊNCIO, Thiago**;

de programação [1] e [5], embora na prática são pouco utilizados. O acompanhamento estatístico no desenvolvimento de software se apresenta apenas nos níveis superiores de maturidade de processo [2], onde são poucas as empresas que têm certificação em tais níveis. Portanto, o uso de métodos estatísticos em programação pressupõe alto grau de maturidade no processo utilizado durante uma competição. A isto se propõe o PSP [2], o qual é aderente aos processos de competição *cf.* [1], [3] e [5]. Como método preditivo, o PSP sugere o uso do *PROxy-Based Estimating* (PROBE) [2]. Este método consiste em dividir um problema em partes conhecidas (*proxies*), para as quais já se tem dados históricos, e a partir dessas informações inferir o tamanho ou dificuldade do *proxy* a ser desenvolvido e por regressão linear estabelecer a estimativa.

METODOLOGIA

O método PROBE será aplicado à problemas de competição de programação de maneira simulada e a precisão da estimativa será comparada ao realizado.

RESULTADOS E DISCUSSÃO

Em competições, problemas *ad hoc* são geralmente resolvidos sem preocupação com a estruturação do código fonte. O PSP entretanto requer que o código siga um padrão de estilo para tornar a contagem das linhas de código (LOCs) mais assertiva. Por isso, adotou-se como categoria de *proxy*, os tipos de problema encontrados em uma competição, no caso deste estudo apenas categoria *problemas ad hoc*. Quando se depara com um problema *ad hoc* em uma competição, ele é conhecido ou desconhecido e isso determinará as categorias consideradas neste estudo.

Para que seja possível usar o PROBE é necessário a coleta de dados históricos cuja correlação entre linhas de código e tempo de desenvolvimento seja de $|r| \geq 0.70$ e que a medida de tamanho (LOC) do *proxy* deve ser relacionada com o esforço (tempo) gasto no desenvolvimento do produto [2]. Para isso foram implementados problemas do tipo *ad hoc* de onde obteve-se o conjunto de dados históricos H com os pares $(l,\ t)$ onde $'l'$ é o número de linhas de código e $'t'$ o tempo necessário para o desenvolvimento. O conjunto $H_1 = \{(23, 29), (26, 39), (32, 53)\}$ apresentou correlação $|r| = 0.98$, habilitando o uso do PROBE para problemas *ad hoc desconhecidos*. O $H_2 = \{(38, 17), (46, 22), (55, 37)\}$ apresentou correlação $|r| = 0.96$, habilitando o uso do PROBE para problemas *ad hoc conhecidos*. Com base nesses dados calculou-se as métricas relativas para estas categoria de problema classificando-as em termos de dificuldade, ver tabela 1.

Tabela 1.- Métricas relativas para os *proxies* '*problema ad hoc conhecido*' e *'problema ad desconhecido*.

LOC x Problema (tamanho)					
Categoria	Muito Simples	Simples	Médio	Difícil	Muito Difícil
Problema *ad hoc* desconhecido	19.1551	22.6352	26.7475	31.607	37.3493
Problema *ad hoc* conhecido	31.6489	38.0771	45.8108	55.1153	66.3097
TEMPO x Problema (tempo)					
Categoria	Muito Simples	Simples	Médio	Difícil	Muito Difícil
Problema *ad hoc* desconhecido	21.4135	28.949	39.1363	52.9084	71.5271
Problema *ad hoc* conhecido	10.8711	16.1553	24.0081	35.678	53.0203

Tais métricas tem caráter individual, ou seja, se aplicam apenas ao competidor que as produziu. Neste sentido, não é possível replicá-las para outros competidores e por outro lado cada competidor deve construir e manter sua própria base de dados históricos para fins táticos. Considere como exemplo uma prova com 300 minutos de duração; este competidor especificamente conseguiria resolver no máximo cerca de: a) 15 problemas ad hoc desconhecidos "muito simples"; b) 13 "médio"; e d) 8 "muito difícil".

Outra diferença no uso do PSP durante competições de programação é o pressuposto que tanto a correlação como as métricas de dificuldade sejam recalculados ao término do projeto, dessa forma o competidor deve anotar seus dados durante a competição e consolidá-los em suas estatísticas gerais após a competição.

A prova de conceito se deu na forma de uma simulação de competição onde foram sorteados três problemas *ad hoc* os quais foram todos classificados pelo competidor como *ad hoc desconhecidos*. Os resultados obtidos são mostrados na Tabela 2.

Tabela 2.- Simulação da competição.

Problema	Categoria	Dificuldade	Tempo de Estimativa	LOC Est. *(tab. 1)*	Tempo Est. *(tab. 1)*	LOC Real	Tempo Real	Precisão	
								LOC	Tempo
WERTYU	desconhecido	Médio	7min	26.7475	39.1363	27	37	99.06%	94.54%
World Cup	desconhecido	Médio	4min	26.7475	39.1363	28	36	95.52%	91.98%
Hanoi	desconhecido	Muito Simples	2min	19.1551	21.4135	19	33	64.88%	99.19%

Por fim, as métricas foram recalculada para incluir problemas *ad hoc desconhecidos* desenvolvidos na simulação, $H_1 = \{(23,29),(26,39),(32,53),(27,37),(28,36),(19,33)\}$ com correlação $|r| = 0.80$. Com isso as métricas relativas para esta categoria foram também recalculadas, as métricas de tamanho para $MR_l' = \{17.8,\ 21.3,\ 25.4,\ 30.5,\ 36.5\}$ e tempo para $MR_t' = \{24.7,\ 30.3,\ 37.1,\ 45.5,\ 55.71\}$ (valores aproximados). O competidor precisa atentar ao fato de que alguns valores obtidos podem se apresentar discrepantes em relação aos valores já coletados, cabe, avaliar quais foram os motivos dessa diferença [2].

CONSIDERAÇÕES FINAIS/CONCLUSÃO

Uma vez que os resultados da prova de conceito foram favoráveis ao uso deste tipo de abordagem, cabe a realização de novos estudos que contemplem esta linha de pesquisa. Entre eles aplicar a metodologia para outros tipos de problemas; coleta de outros tipos de dados históricos; e utilização do Team Software Process.

REFERÊNCIAS

[1] HALIM, Steven. HALIM, F. (2015). *Competitive programming 3: The New Lower Bound*. National University of Singapore.

[2] HUMPHREY, Watts. *PSP: A self-Improvement Process for Software Engineers*. Massachusetts: Pearson, 2005.

[3] PARK, Jaehyun. *CS 97SI: Introduction to Programming Contests*. Stanford. Disponível em: http://web.stanford.edu/class/cs97si/01-introduction.pdf. Acessado em: 27/06/2017.

[4] KNUTH, Donald. *The art of computer programming: Fundamental Algorithmn and Sorting and Searching*. Addison Wesley Professional, reimpressão, 1997 e 2003.

[5] SKIENA, Steven. REVILLA, Miguel. *Programming Challenges: The programming contest training manual*. New York: Springer-Verlag, 2003.