Estimativa de Esforço em Teste de Software: Modelos, Fatores e Incertezas

Thiago Silva-de-Souza[1], Victor Vidigal Ribeiro[1], Guilherme Horta Travassos[1]

[1] Programa de Engenharia de Sistemas e Computação (PESC), Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia (COPPE), Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro-RJ, Brasil
{thiagosouza, vidigal, ght}@cos.ufrj.br

Abstract. Estimar esforço é uma atividade crítica em Teste de Software. Diversos modelos têm sido propostos na literatura técnica para apoiar tal atividade. Diante deste cenário, este trabalho apresenta os resultados de um estudo secundário que identificou fatores de influência do esforço do teste de software e modelos de estimativa de esforço que fazem uso destes fatores. Os modelos e fatores identificados não se mostram genericamente adequados devido a variabilidade dos projetos. Além disso, as evidências sobre a falta de consenso sobre o que é teste de software e o que é esforço de teste tornam a escolha de qualquer um destes modelos de estimativa uma tarefa arriscada e propensa a erro.

Keywords: Teste de software, estimativa de esforço, fatores de influência.

1 Introdução

Teste de Software é uma das atividades mais importantes do ciclo de vida de um projeto de software. Seu principal objetivo é avaliar a qualidade de um produto de software através de falhas reveladas em sua execução controlada. O teste de software tem recebido atenção contínua da indústria, como pode ser observado através do surgimento de equipes independentes de teste em diversas organizações e a proliferação de companhias especializadas em prestar serviços de teste de software – conhecidas como "Fábricas de Teste" [15].

Testar software demanda esforço. Neste sentido, estimar o esforço dos testes de software é uma atividade importante e crítica para o projeto. A estimativa do esforço é necessária para apoiar os gerentes de testes de software no planejamento dos projetos. A estimativa de esforço apoia a tomada de decisão sobre a alocação de recursos, definição do cronograma de teste, cálculo do custo financeiro do projeto e, até mesmo, ser usada como critério de seleção de casos ou procedimentos de teste a serem executados ou automatizados [16].

Diferentes modelos de estimativa de esforço em teste de software têm sido propostos ao longo dos anos. Os primeiros modelos, tais como COCOMO [3] consideram o esforço em teste como parte do esforço de desenvolvimento. Capers Jones [8] sugere que a estimativa de esforço em teste (expressa inicialmente em números de casos de

teste) seja obtida a partir de uma relação com o tamanho funcional do software (em pontos por função), e convertida em pessoas-hora através de um fator de conversão obtido de dados históricos dos projetos.

Apesar da existência de diferentes modelos relacionados a estimativa de esforço em teste de software, algumas questões ainda afetam o resultado das estimativas obtidas e, consequentemente, podem impor riscos aos projetos:

- o conceito de esforço não é compreendido por todos da mesma maneira [9] – muitos trabalhos tratam esforço como sinônimo de custo ou tempo de projeto, podendo levar a interpretações equivocadas;
- a estimativa do esforço do teste leva em consideração características inerentes ao desenvolvimento do software, sem evidência de que tais características efetivamente afetam o esforço de teste – os fatores de influência (*cost drivers*) do teste de software são claramente diferentes dos fatores que influenciam outras atividades de desenvolvimento de software [11]. É necessário, portanto, evidenciar que fatores de projeto afetam o esforço de teste e avaliar como eles devem ser considerados para estimar o esforço dos testes;
- grande parte dos modelos específicos para teste de software, por exemplo os propostos por Aranha & Borba [1] e Xiaochun *et al*. [16], estimam apenas o esforço da execução de testes – entretanto, um processo de teste contempla outras atividades, tais como planejamento, projeto (*design*) e análise dos resultados [6]. Desta forma, é preciso considerar o esforço inerente a essas atividades ao estimar o esforço total em projetos de teste.

Sendo assim, este trabalho apresenta os resultados de um estudo secundário realizado com o objetivo de identificar, na literatura técnica, o entendimento sobre esforço de teste de software, os modelos de estimativa de esforço de Teste de Software e os fatores de influência de esforço utilizados por estes modelos. Um estudo secundário revisa os estudos primários relacionados a uma questão de pesquisa particular com o objetivo de integrar/sintetizar as evidências relacionadas [2]. O restante deste artigo está organizado da seguinte maneira: a seção 2 descreve o protocolo utilizado nesta revisão; a seção 3 apresenta os principais resultados encontrados; a seção 4 discute o resultados, ressaltando as lacunas deixadas pelos trabalhos analisados; por fim, a seção 5 conclui o trabalho e apresenta sugestões de trabalhos futuros.

2 Protocolo do Estudo

Esta seção apresenta os principais itens do protocolo de estudo secundário: as questões de pesquisa e as estratégias de busca, seleção e extração dos artigos. As questões de pesquisa, definidas em função do cenário apresentado na seção 1, são as seguintes:

- **Q1**: O que se entende por esforço no teste de software e como ele pode ser medido?
- **Q2**: Que fatores influenciam o esforço de teste de software?
    - **Q2.1**: Que modelos utilizam esses fatores?
        - **Q2.1.1**: Tais modelos são baseados em que tipos de técnicas?
        - **Q2.1.2**: Tais modelos são adequados para qual contexto de teste?
        - **Q2.1.3**: Que tipos de estudo têm sido realizados para avaliar tais modelos?

Uma *string* de busca, representada na **Figura 1**, foi definida com base na abordagem PICO [14]. As palavras-chave que compõem essa *string* foram identificadas a partir do conhecimento prévio dos pesquisadores e derivadas de sete artigos de controle identificados através de uma busca *ad hoc* (lista de artigos de controle disponível em *http://lens-ese.cos.ufrj.br/te/list.pdf*). Tais artigos foram considerados de controle porque apresentam respostas para as questões de pesquisa apresentadas e serviram para calibrar a *string* de busca. A identificação dos artigos relacionados foi realizada através da execução da *string* de busca nas máquinas de busca Scopus, IEEEXplore, Engineering Village e Web of Science. A escolha dessas máquinas de busca foi motivada pela cobertura que elas oferecem. Por exemplo, a Scopus indexa artigos de diferentes fontes, incluindo ACM e IEEE. Além disso, tais máquinas de busca são conhecidas pela estabilidade e interoperabilidade com diferentes sistemas de referência.

(("software testing" OR "software test" OR "testing of systems" OR "system test" OR "system testing" OR "software verification" OR "software validation" OR "verification & validation" OR "software evaluation" OR "software quality" OR "test activities" OR "testing activities" OR "test activity" OR "testing activity" OR "test planning" OR "test design" OR "test automation" OR "automated test" OR "test execution" OR "manual test" OR "test case" OR "test specification" OR "testing process" OR "testing project" OR "test project") **AND** ("software test estimation" OR "effort estimation" OR "cost estimation" OR "estimate the cost" OR "effort prediction" OR "cost prediction" OR "effort measurement" OR "cost measurement" OR "execution effort" OR "time estimation") **AND** ("factor" OR "variable" OR "predictor" OR "indicator" OR "driver" OR "measure" OR "model" OR "approach" OR "method" OR "process" OR "technique" OR "metric"))

Figura 1. String de busca utilizada.

Após a execução das buscas, foram aplicados os critérios de inclusão (I1 e I2 e I3 e I4 e (I5 ou I6) e I7) e de exclusão (E1 ou E2 ou E3 ou E4 ou (E5 e E6) ou E7 ou E8) representados na **Tabela 1**.

Tabela 1. Critérios de seleção dos artigos.

Critérios de Inclusão	Critérios de Exclusão
(I1) Estar disponível na Web	(E1) Não estar disponível
(I2) Estar escrito em Inglês	(E2) Não estar escrito em Inglês
(I3) Tratar de Teste de Software	(E3) Não tratar de Teste de Software
(I4) Abordar estimativa de esforço	(E4) Não abordar estimativa de esforço
(I5) Apresentar fatores que exercem influência no Teste de Software	(E5) Não apresentar fatores que exercem influência no esforço de Teste de Software
(I6) Apresentar ou avaliar modelos para estimar o esforço no Teste de Software	(E6) Não apresentar ou avaliar modelos para estimar o esforço no Teste de Software
(I7) Apresentar qualquer avaliação experimental ou prova de conceito	(E7) Não apresentar qualquer avaliação experimental ou prova de conceito
	(E8) Utilizar bases públicas de projetos, tais como Promise e ISBSG

A existência do critério E8 se justifica pelas limitações no uso de bases de projetos multi-organização devido a sua heterogeneidade e pelo problema em usar uma base de dados multi-organizacional para produzir modelos para organizações que não contribuem com tal base [5]. Além disso, derivar modelos a partir de bases de dados muito grandes e com dados redundantes ou irrelevantes é mais difícil que derivar um modelo a partir de uma base de dados homogênea e limitada a um conjunto de atributos altamente preditivos.

A **Tabela 2** apresenta os resultados da busca e da seleção de artigos considerando os critérios de seleção da **Tabela 1**. Para aumentar a cobertura da busca foi aplicada a técnica de *snowballing* (*forward* e *backward* de um nível) [7] através do Google Scholar, resultando na inclusão de cinco artigos que estavam fora das bases cobertas pelas máquinas de busca citadas anteriormente. A lista de artigos incluídos está disponível em *http://lens-ese.cos.ufrj.br/te/list.pdf*. A extração das informações dos artigos foi apoiada por um formulário de extração (disponível em *http://lens-ese.cos.ufrj.br/te/form.pdf*), definido com base nas questões de pesquisa apresentadas. Os artigos foram, então, distribuídos aleatoriamente para extração entre os dois primeiros autores. Posteriormente, cada formulário de extração foi revisado em conjunto por todos os autores a fim de garantir a consistência da forma de extração e de apresentação dos dados.

Tabela 2. Resultados da busca e seleção.

Status dos Artigos	Quantidade
Retornados pelas bases (com duplicatas)	834
Retornados pelas bases (sem duplicatas)	571
Controle presentes nas bases	6
Controle fora das bases	1
Excluídos	544
Incluídos	21
Incluídos por *snowballing*	5
Total de artigos incluídos	**33**

3 Resultados

Esta seção apresenta os resultados obtidos através do estudo secundário, organizados pelas questões de pesquisa apresentadas no protocolo de estudo.

3.1 Q1: O que se entende por esforço no teste de software e como esforço pode ser medido?

A palavra "esforço" é definida pelo dicionário de Cambridge como "a atividade física ou mental necessária para alcançar algum objetivo" [4]. Entretanto, no contexto deste estudo se fez necessário compreender o entendimento de esforço entre os pesquisadores de modelos de estimativa de esforço em teste de software. Desta forma, procurou-se identificar nos artigos analisados qualquer definição sobre esforço em teste de software, bem como quais unidades de medida são utilizadas para representá-lo.

Para a maioria dos 33 trabalhos analisados, o conceito de esforço é definido de forma tácita, sendo que apenas quatro artigos apresentam alguma definição explícita do que é esforço em teste de software. Já em relação à unidade de medida de esforço, é possível notar na **Tabela 3** que as unidades de medida do tipo pessoa-tempo (e.g. homem-hora e pessoa-mês) são as mais frequentes para mensurar esforço (há trabalhos que indicam mais de uma medida).

Tabela 3. Distribuição das unidades de medida de esforço.

Unidade de Medida de Esforço	Ocorrências
<Pessoa>-<tempo>	24
Tempo	9
Tamanho	1
% Esforço de desenvolvimento	1
Não especificada	4

3.2 Q2: Que fatores influenciam o esforço de teste de software?

A correta definição dos fatores que influenciam o esforço de teste de software em cada tipo de projeto pode apoiar a construção de modelos de estimativa mais realistas. Sendo assim, procurou-se identificar nos artigos incluídos os fatores utilizados pelos modelos para estimar o esforço em teste de software. Parte dos trabalhos apresenta explicitamente os fatores utilizados, enquanto outros realizam definições implícitas através das equações dos modelos. Os fatores de influência identificados foram, então, organizados a partir de seis diferentes categorias: **1) software sob teste** (ou *system under testing* – SUT); **2) equipe**; **3) processo de teste**; **4) projeto de teste**; **5) ambiente**; e **6) *testware*** (artefatos de teste produzidos). Tais categorias emergiram das classificações apresentadas nos artigos para os fatores de influência.

As ocorrências de cada fator de influência foram contabilizadas e agrupadas de acordo com a definição de cada fator indicada nos respectivos trabalhos. Entretanto, como nem todos os trabalhos apresentavam definições explícitas sobre cada fator, os autores deste estudo acordaram o significado do fator com base no seu nome e contexto apresentado no artigo. A **Tabela 4** apresenta o número de ocorrências de cada fator de influência identificado de acordo com as categorias de esforço identificadas.

Pode-se observar na **Tabela 4** que a categoria SUT apresenta mais tipos de fatores de influência (13), dentre os quais os fatores mais frequentes são complexidade e tamanho do software. Em geral, o tamanho do SUT tem sido representado em KLOC ou pontos por função. Já a complexidade tem sido representada de diversas formas, tais como a complexidade dos requisitos, a complexidade do código e a complexidade de integração de sistemas. Vale ressaltar que um mesmo modelo pode apresentar mais de uma característica relacionada a mesmo fator de influência.

Os fatores de influência mais frequentes relacionados com a equipe dizem respeito à eficiência e experiência. A eficiência geralmente é indicada nos trabalhos como produtividade ou como capacidade da equipe, enquanto que a experiência costuma ser tratada separadamente como experiência em teste, experiência no domínio e experiência com a tecnologia de programação.

Em relação aos fatores relacionados ao processo de teste, nota-se que as restrições impostas pelo processo de teste (e.g. procedimentos requeridos devido à política organizacional) aparecem com maior frequência. Já em relação ao projeto de teste, foram identificados apenas dois fatores de influência: cronograma de teste requerido e teste *multisite*.

Os fatores de influência com maior ocorrência relacionados ao ambiente são a complexidade do ambiente de teste e o uso de ferramentas de teste. Em geral, a complexidade do ambiente de teste é referenciada nos artigos analisados como complexi-

dade de configuração ou de integração do ambiente de teste. Quanto ao uso de ferramentas de teste, os artigos, em geral, consideram não apenas ferramentas de apoio a execução dos testes, mas também aquelas que apoiam o planejamento dos testes e a gestão de defeitos.

Tabela 4. Ocorrências de fatores de influência.

Categoria de Fatores	Fator de Influência	Ocorrências
SUT	Complexidade do software	38
	Tamanho	20
	Desempenho requerido	9
	Confiabilidade requerida	9
	Segurança requerida	8
	Tipo de sistema	8
	Testabilidade	6
	Estabilidade de requisitos	5
	Portabilidade requerida	4
	Maturidade da tecnologia	3
	Volume de dados	3
	Usabilidade requerida	3
	Esforço gasto em outras atividades	3
		Subtotal: 119
Equipe	Eficiência da equipe	13
	Experiência com teste	11
	Experiência com a tecnologia de programação	11
	Experiência no domínio	10
	Alocação apropriada de recursos	5
	Coesão da equipe	3
	Continuidade da equipe	3
		Subtotal: 56
Processo de Teste	Restrições do processo	9
	Tipo de teste	6
	Maturidade do processo de teste	4
	Ciclos de teste	3
		Subtotal: 22
Projeto de Teste	Cronograma de teste requerido	5
	Teste *multisite*	3
		Subtotal: 8
Ambiente	Complexidade do ambiente de teste	23
	Uso de ferramentas de teste	15
	Dependência de rede	7
	Complexidade do ambiente de desenvolvimento	6
	Desempenho do hardware	3
		Subtotal: 54
Testware	Tamanho do teste	36
	Complexidade de execução do teste	32
	Reusabilidade de artefatos de teste	9
	Defeitos encontrados	5
		Subtotal: 82
		Total: 341

Finalmente, os fatores de influência com mais ocorrências relacionados ao *testware* foram o tamanho do teste e a complexidade de execução dos casos de teste. O tamanho do teste geralmente é influenciado pela quantidade de casos de teste e pela quantidade de passos de cada caso de teste. A complexidade de execução apresentada nos

artigos costuma levar em consideração os tipos de componentes a serem testados e a necessidade de acessar arquivos ou recursos multimídia. Vale ressaltar que estes dois fatores estão entre os três mais frequentes de todo o estudo.

3.3 Q2.1: Que modelos utilizam esses fatores?

A caracterização dos modelos que utilizam os fatores de influência identificados se dá pelas questões Q2.1.1, Q2.1.2 e Q2.1.3, descritas nas seções 3.4, 3.5 e 3.6.

3.4 Q2.1.1: Tais modelos são baseados em que tipos de técnicas?

Uma das preocupações deste trabalho foi como classificar os modelos de acordo com o tipo de técnica de estimativa. Há diversos trabalhos que propõem formas de classificação de modelos de estimativa de esforço [9] [12]. Entretanto, tais classificações não se mostraram adequadas para os trabalhos aqui analisados. Sendo assim, decidiu-se por deixar as categorias emergirem dos próprios trabalhos sob análise, evitando uma possível perda de informação relevante. A **Figura 2**, portanto, representa a distribuição de frequência dos modelos analisados por tipo de técnica de estimativa.

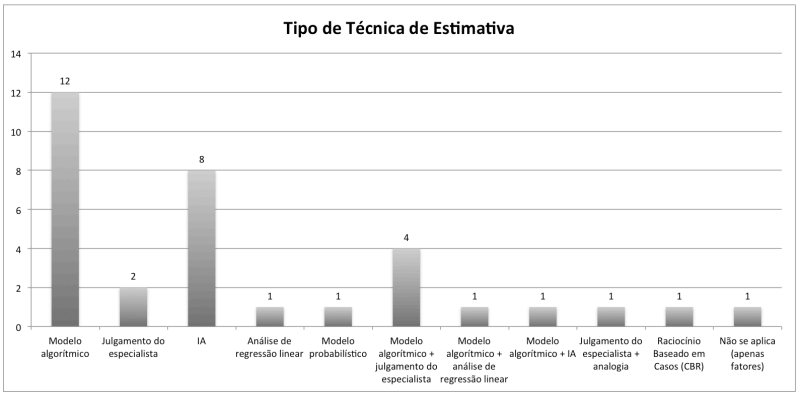

Figura 2. Distribuição dos modelos por tipo de técnica de estimativa.

É possível observar que a maior parte dos modelos analisados segue uma abordagem algorítmica – dos 33 trabalhos, 12 apresentam modelos estritamente algorítmicos e outros seis apresentam modelos algorítmicos associados a outras técnicas, em especial julgamento do especialista. Cabe destacar que nove modelos são baseados (total ou parcialmente) em técnicas de inteligência artificial (IA), dos quais quatro utilizam os mesmos dados de projetos, variando apenas a técnica de IA.

3.5 Q2.1.2: Tais modelos são adequados para qual contexto de teste?

O contexto de teste foi caracterizado neste estudo por cinco dimensões: nível, tipo, técnica, fase e forma de execução de testes, de acordo com a norma IEEE 829-2008 [6]. Na **Figura 3** pode-se notar que a maior parte dos trabalhos não contextualiza claramente cada dimensão de teste para a qual seu modelo é proposto. Em todas as dimensões de teste consideradas a falta de especificação se mostrou mais frequente.

Apenas 1/3 dos artigos analisados indica explicitamente o nível de teste coberto pelo modelo, dentre os quais percebe-se que o teste em nível de sistema é o mais frequente, conforme mostra a **Error! Reference source not found.** (a). A **Figura 3** (b) indica que o tipo de teste funcional é mais frequente do que o teste não-funcional. Em relação à técnica de teste, a **Figura 3** (c) mostra que a técnica funcional (caixa preta) é mais frequente que a técnica estrutural (caixa branca). Quanto à fase de teste, a **Figura 3** (d) mostra que a fase de execução é a mais frequente entre as fases de teste indicadas nos modelos que caracterizam tal dimensão.

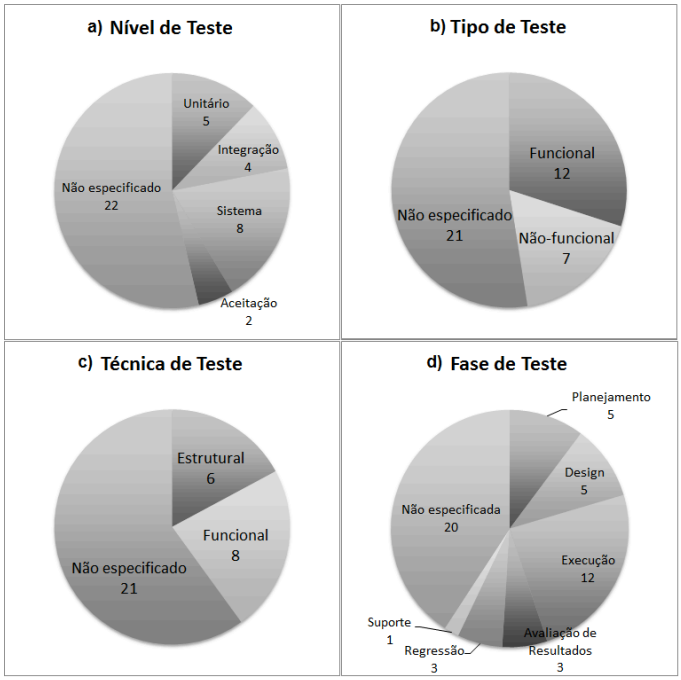

Figura 3. Distribuição de níveis, tipos, técnicas e fases de teste entre os trabalhos analisados.

Cabe ressaltar também que, dos 33 trabalhos analisados, apenas sete indicam claramente qual a forma de execução dos testes (manual ou automatizada). Desses sete trabalhos, quatro cobrem apenas a execução manual, um compreende apenas a execução automatizada e dois compreendem ambas as formas de execução.

3.6 Q2.1.3: Que tipos de estudo têm sido realizados para avaliar tais modelos?

Os estudos relatados nos trabalhos foram classificados de acordo com seu tipo. A **Tabela 5** mostra que os tipos de estudo mais frequentes são estudo de caso, *survey* e prova de conceito (ambos com 11 ocorrências). Vale salientar que há artigos que apresentam mais de um estudo (e.g. um *survey* e dois estudos de caso).

Para tornar mais precisa a caracterização dos estudos o perfil dos participantes também foi considerado. Observou-se que os estudos utilizam com mais frequência projetos da indústria (23 ocorrências), seguidos de projetos fictícios (oito ocorrências), profissionais (quatro ocorrências) e, por fim, estudantes (uma ocorrência).

Tabela 5. Frequência dos estudos por tipo de estudo.

Tipo de Estudo	Ocorrências
quasi-Experimento	2
Estudo de caso	11
Survey	11
Simulação	5
Prova de conceito	11

4 Discussão

Analisando os resultados apresentados é possível constatar que aparentemente não há consenso na área de teste de software sobre o conceito de esforço e não há uma visão homogênea sobre o que é teste de software. Os trabalhos analisados tratam, em geral, estes conceitos de forma tácita e os consideram de entendimento comum. Além disso, as poucas definições apresentadas sobre esforço em teste se mostram superficiais e ambíguas. Por exemplo, dos quatro trabalhos, três definem recursivamente o esforço em teste de software. Os trabalhos de Mizuno *et al*. [13] e Kumari & Sharma [10] incluem no esforço de teste o esforço despendido com atividades de depuração – atividades de depuração correspondem à identificação e correção de defeitos, portanto são consequência do teste e representam retrabalho. Entendemos que, neste caso, o esforço de depuração deve incrementar o esforço de construção, e não o de teste. Consideramos esforço de teste o conjunto de ações necessárias para realizar as atividades de planejamento, projeto, execução e análise de resultados de testes, conforme a norma IEEE 829-2008 [6]. Essa falta de entendimento do que é esforço em teste é ainda evidenciada pelas diferentes unidades de medida utilizadas. Apesar de a maioria dos trabalhos utilizar medidas do tipo pessoa-tempo, ainda existem trabalhos que consideram equivocadamente apenas tamanho ou tempo como medidas de esforço.

Quanto aos fatores de influência, observou-se que muitos trabalhos não os descrevem claramente e não informam como medi-los, o que torna seu entendimento passível de múltiplas interpretações. Outro ponto crítico observado é a falta de indicação da influência do fator, isto é, se aumenta ou diminui o esforço e em que proporção.

A maior parte dos modelos analisados segue uma abordagem algorítmica, apresentando em geral equações compostas por fatores que não foram avaliados experimentalmente – a avaliação foi realizada apenas sobre o modelo, com diversos fatores incluídos. Desta forma, não é possível determinar quais fatores realmente exercem influência sobre o esforço, porém, também não é possível dissociá-los de seus modelos. Esta incerteza pode levar gerentes de testes a tomar decisões com base em fatores que não exercem influência efetiva sobre o esforço de teste.

Os modelos de estimativa analisados não indicam as dimensões de teste para as quais foram concebidos. Cada dimensão de teste possui fatores de influência específicos ou em diferentes proporções. Por exemplo, a complexidade de execução de casos de teste pode ter diferentes medidas de esforço para teste manual e para teste automatizado. A falta de indicação dessas dimensões talvez se explique pela falta de consenso a respeito do que é a atividade de teste de software.

Quanto aos estudos conduzidos para avaliar os modelos, observa-se uma tendência por estudos com baixo nível de controle, tais como *surveys* e provas de conceito, o

que compromete a confiança nas evidências produzidas. Estudos com maior nível de controle que usualmente produzem resultados mais robustos não são observados.

5 Conclusão

Este trabalho apresentou os resultados de um estudo secundário a respeito de modelos de estimativa de esforço em teste de software e os fatores de influência que os compõem. O estudo foi norteado por questões de pesquisa que visaram a caracterizar o que se entende por esforço em teste de software, quais fatores influenciam o esforço desta atividade e quais modelos utilizam tais fatores.

Pode-se concluir que os modelos e fatores identificados não se mostram adequados a projetos de qualquer natureza, devido a variabilidade dos projetos. Além disso, as evidências sobre a falta de consenso sobre o que é teste de software e o que é esforço de teste tornam a escolha de qualquer modelo uma tarefa arriscada e propensa a erros.

Com o objetivo de avaliar os resultados apresentados sobre os modelos e fatores, bem como identificar novos fatores e dimensões de esforço, será conduzido futuramente um *survey* com especialistas.

Referências Bibliográficas

1. Aranha, E. & Borba, P. (2007). An estimation model for test execution effort. Proceedings of the ESEM 2007, (p. 107-116). Madrid, Spain.
2. Biolchini, J. C., Mian, P., Natali, A., Conte, T. & Travassos, G. H. (2007). Scientific research ontology to support systematic review in software. Advanced Engineering Informatics, 21 (2), 133-151.
3. Boehm, B. (1981). Software Engineering Economics. Englewood Cliffs: Prentice-Hall.
4. Cambridge University Press. (2014). Cambridge Dictionaries Online. Retrieved 2014, from Cambridge Dictionaries Online: http://dictionary.cambridge.org/dictionary.
5. Fernández-Diego, M. & González-Ladrón-de-Guevara, F. (2014). Potential and limitations of the ISBSG dataset in enhancing software engineering research: A mapping review. Journal of Information and Software Technology, 56 (6), 527-544.
6. IEEE Computer Society. (2008). IEEE Standard for Software and System Test Documentation. IEEE Computer Society. New York, NY: IEEE Computer Society.
7. Jalali, S., & Wohlin, C. (2012). Systematic Literature Studies: Database Searches vs. Backward Snowballing. Proceedings of the ESEM 2012 (p. 29-38). New York: ACM.
8. Jones, C. (1996). Applied Software Measurement (2nd ed.). McGraw-Hill.
9. Jorgensen, M., & Shepperd, M. (2007). A Systematic Review of Software Development Cost Estimation Studies. IEEE Transactions on Software Engineering, 33 (1), 33-53.
10. Kumari, A., & Sharma, A. (2013). Test effort estimation in regression testing. Proc. 2013 International Conference in MOOC, Innovation and Technology in Education (p. 343-348).
11. Lu, Y.-f., & Yin, Y.-f. (2013). A New Constructive Cost Model for Software Testing Project Management. Proceedings of the 19th IEEM (p. 545-556). Springer.
12. Mendes, E. (2007). Cost Estimation Techniques for Web Projects. Hershey: IGI Global.
13. Mizuno, O., Shigematsu, E., Takagi, Y. & Kikuno, T. (2002). On estimating testing effort needed to assure field quality in software development. Proceedings of the ISSRE 2002, (p. 139-146). Annapolis.
14. Pai, M., McCulloch, M., & Gorman, J. D. (2004). Systematic Reviews and metaanalyses: An illustrated, step-by-step guide. The National Medical Journal of India, 17 (2).
15. SOFTEX. (2011). Guia de Implementação – Parte 10: Implementação do MR-MPS em organizações do tipo Fábrica de Teste. SOFTEX.
16. Xiaochun, Z. et al. (2008). Estimate Test Execution Effort at an Early Stage: An Empirical Study. Proc. of the International Conference on Cyberworlds 2008, (p. 195-200).