Estimativa de Esforço em Atividades de Manutenção de Software: Um Mapeamento Sistemático

Kaliane L. Viesseli[1], **Arielyn P. Silva**[1], **Gustavo Santos**[1]

[1]COENS – Universidade Tecnológica Federal do Paraná (UTFPR)
Estrada para Boa Esperança, km. 04 – Zona Rural, Dois Vizinhos - PR, 85660-000

{kaliane, arielyn}@alunos.utfpr.edu.br, gustavosantos@utfpr.edu.br

Abstract. *During the software lifecycle, good effort estimation allows teams to make decisions on how development activities will proceed, on the feasibility of new changes, and whether the project will be delivered according to deadlines. In this paper, we describe a systematic mapping to identify evidence with respect to metrics, approaches and frameworks for calculating effort estimation during the software maintenance phase. We analyzed 521 studies and we selected 17 primary studies; most approaches use metrics and historical data do estimate effort for new activities. However, only one study proposed a framework, which emphasizes the importance of proposing tools to calculate effort estimation for maintenance activities.*

Resumo. *Durante o ciclo de vida do software, o correto levantamento de estimativas de esforço permite que a equipe tome decisões sobre como será o andamento das atividades de desenvolvimento e manutenção, qual a viabilidade de novas alterações, e se estas serão entregues de acordo com os prazos. Neste artigo, é apresentado um mapeamento sistemático com o objetivo de identificar evidências na literatura com relação a métricas e abordagens para o cálculo de estimativa de esforço durante a fase de manutenção de software. Foram analisados 521 estudos e selecionados 17 estudos primários; a maioria das abordagens utiliza de métricas ou dados históricos para estimativa de novas atividades, e apenas um framework foi proposto, o que ressalta a importância de gerar novas ferramentas que facilitem a geração das novas estimativas.*

1. Introdução

A manutenção de software abrange os processos de modificação de um sistema após a sua entrega, sejam estes para correções de falhas, melhorias de desempenho ou adaptações do sistema para um ambiente diferente. A manutenção é considerada a fase mais custosa do ciclo de vida de um software, atingindo até 80% dos custos totais de um projeto, e cerca de 75% dos custos são dedicados em melhorias [Erlikh 2000].

As atividades de manutenção em maioria são reativas; isto é, erros são identificados no sistema em uso, melhorias não planejadas no início do projeto são requisitadas pelos clientes, e mudanças na plataforma exigem atualizações no sistema. Independente da atividade, as empresas demandarão esforço e tempo para realizar as alterações necessárias, considerando o estado atual das equipes, a complexidade da atividade, o nível de experiência com o sistema, entre outros fatores. Para isso, as estimativas são responsáveis por determinar quanto de esforço será empregado, qual o custo, qual será o cronograma e o escopo do projeto [Vazquez et al. 2013].

O presente trabalho abordará em específico a estimativa de esforço, a qual almeja identificar qual o tempo necessário para a conclusão de uma atividade. Neste sentido, existem muitos fatores que podem dificultar na geração de uma estimativa mais precisa, como por exemplo requisitos imprecisos e mal detalhados, falta de conhecimento dos estimadores e nova atividade muito diferente do que já foi realizado no projeto, não sendo possível utilizar atividades semelhantes para realizar uma estimativa por analogia.

O objetivo deste trabalho é identificar as abordagens propostas na literatura sobre estimativa de esforço, sumarizar as características principais dos trabalhos encontrados e, consequentemente, situar a atual pesquisa e sugerir novas áreas de investigação para futuras pesquisas. Para realização desse objetivo, foi conduzido um Mapeamento Sistemático (MS) [Kitchenham 2004] em busca de trabalhos relevantes publicados em conferências e periódicos de impacto.

O restante do artigo está organizado da seguinte forma: a Seção 2 descreve o protocolo de pesquisa e condução do MS. A Seção 3 apresenta os resultados e lições aprendidas com o estudo. A Seção 4 discute limitações e ameaças à validade. E, por fim, a Seção 5 apresenta as considerações finais.

2. Mapeamento Sistemático

Com o objetivo de identificar métodos, métricas ou ferramentas de estimativa de esforço utilizados durante a manutenção de software, foi realizado um MS, o qual visa identificar, interpretar e avaliar todas as pesquisas pertinentes à questão de pesquisa definida [Kitchenham 2004]. Este MS foi realizado seguindo quatro passos: (i) definição da questão de pesquisa (QP), (ii) pesquisa dos estudos primários relacionados ao tema, (iii) triagem desses estudos e (iv) extração de dados. Para a condução deste MS, a seguinte QP foi definida:

- **QP:** Quais métodos, métricas ou ferramentas de estimativa de esforço têm sido aplicadas à manutenção de software?

Esta QP visa identificar diferentes abordagens para estimativa de esforço, especificamente aplicadas para apoiar atividades referentes à manutenção de software. Durante esta fase, a complexidade do sistema tende a aumentar, a qualidade do código decai e novas alterações irão demandar mais esforço para compreensão do estado atual do sistema. Consequentemente, estimativas realizadas com base nas primeiras versões do projeto não refletem o real esforço para manutenção.

A partir da QP, foram identificadas as palavras chaves relacionadas ao tema, mais especificamente os termos "estimativa de esforço" e "manutenção de software". A *string* de busca foi obtida pela combinação dos sinônimos em inglês destes termos, obtendo então a seguinte *string*: (***"effort estimation" AND "software maintenance"***).

Posteriormente, quatro bases de dados foram selecionadas para extração dos estudos primários, sendo elas: *ACM Digital Library*, *IEEE Xplore*, *Scopus* e *Springer*.[1]. Por meio da *string* de busca, são buscados automaticamente estudos em conferências e periódicos de impacto na área.

[1] `http://dl.acm.org/` — `http://ieeexplore.ieee.org/` — `http://scopus.com/` — `http://link.springer.com/`

Para a triagem dos estudos obtidos na etapa anterior, foram definidos critérios de inclusão e exclusão. Ambos são importantes para classificar os estudos mais relevantes e que possam contribuir com a QP deste trabalho. Os **Critérios de Inclusão (CI)** agregam: **CI_1**: estudos primários que abordem alguma metodologia ou ferramenta propondo estimativas de esforço voltadas para a área de manutenção de software; e **CI_2**: estudos primários em que a estimativa de esforço seja voltada a tarefas ou a pequenos projetos.

Quanto aos **Critérios de Exclusão (CE)**, são definidos a seguir: **CE_1**: estudos primários que não sejam *full paper* ou *short paper*; **CE_2**: estudos primários que estejam escritos em outro idioma que não seja Inglês ou Português; **CE_3**: estudos primários incompletos e que não possuem avaliação dos métodos; **CE_4**: estudos primários indisponíveis para *download*; **CE_5**: estudos primários que são versões anteriores de um estudo mais completo sobre a mesma investigação.

A condução deste MS é apresentado na Figura 1. As buscas automáticas nas bases de pesquisa foram realizadas no período de outubro a novembro de 2019.

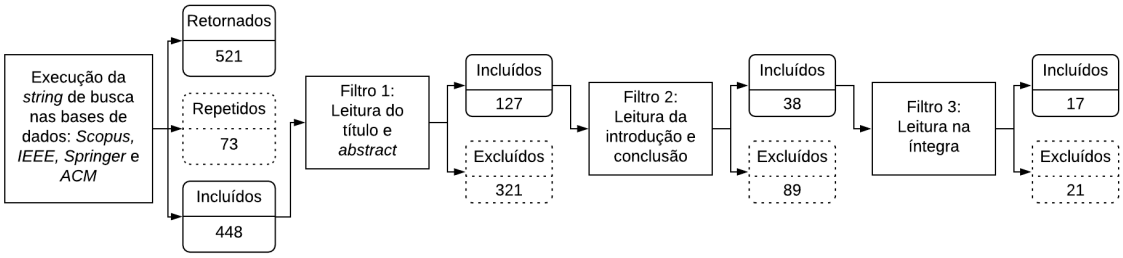

Figura 1. Processo de condução do MS

Após a execução da *string* de busca nas bases selecionadas, foram retornados 521 estudos, sendo 19 estudos na *ACM*, 204 no *IEEE*, 95 no *Scopus*, e 203 no *Springer*. A partir dos resultados obtidos, identificou-se 73 artigos repetidos, os quais foram excluídos do mapeamento, assim restando 448 estudos a serem analisados. Na próxima fase, foram aplicados os critérios de inclusão e exclusão a partir da leitura do título e resumo dos estudos selecionados, sendo incluídos 127 estudos e excluídos 321 estudos. Em seguida, os mesmos critérios foram aplicados, desta vez a partir da leitura da introdução e conclusão dos estudos; foram incluídos 38 estudos e excluídos 89 estudos nesta fase. Por fim, o último filtro foi baseado na leitura integral dos estudos, dos quais 17 estudos foram incluídos e 21 foram excluídos.

É importante destacar que, para melhor compreensão do contexto de cada estudo primário, foram criadas três categorias para classificação desses estudos conforme os objetivos da QP. As categorias são:

- **C_1 – Métricas:** reúne estudos que sugerem o uso de métricas, *e.g.*, linhas de código e pontos de função, ou fórmulas para o cálculo da estimativa de esforço;
- **C_2 – Métodos:** reúne estudos que propõem um método, técnica ou abordagem para o cálculo da estimativa de esforço;
- **C_3 – Ferramentas:** reúne estudos que apresentam uma ferramenta, protótipo, sistema, software, ou *framework* para o cálculo de estimativa de esforço.

Após a leitura na íntegra, foi possível associar alguns estudos a mais de uma categoria. Os conjuntos de categorias observadas foram: C_1 e C_2; e C_2 e C_3.

3. Resultados

A partir da execução do MS, foram selecionados 17 estudos que apresentavam algum método para estimativa de esforço na área de manutenção de software. Considerando o meio de publicação, nota-se que a maioria dos estudos foi publicado em conferências, correspondendo a 58,8% (10/17), já os demais foram publicados em periódicos – 29,4% (5/17), simpósio – 5,9% (1/17) e workshop – 5,9% (1/17).

Quanto ao ano de publicação, pode-se observar que essa área possui estudos ao passar dos anos, tendo artigos desde 1997 até 2019, sendo que os anos que obtiveram mais artigos foram 2008, 2011 e 2018, possuindo dois estudos cada ano. Utilizando como base o país de origem do primeiro autor, os 17 artigos estão distribuídos em nove países diferentes, sendo Índia o país com mais estudos nessa área, totalizando quatro artigos; Estados Unidos e Brasil produziram três estudos cada, e a Coréia do Sul publicou dois artigos. Nas Figuras 2 e 3 são apresentadas as distruibuições de estudos por ano de publicação e país do primeiro autor, respectivamente.

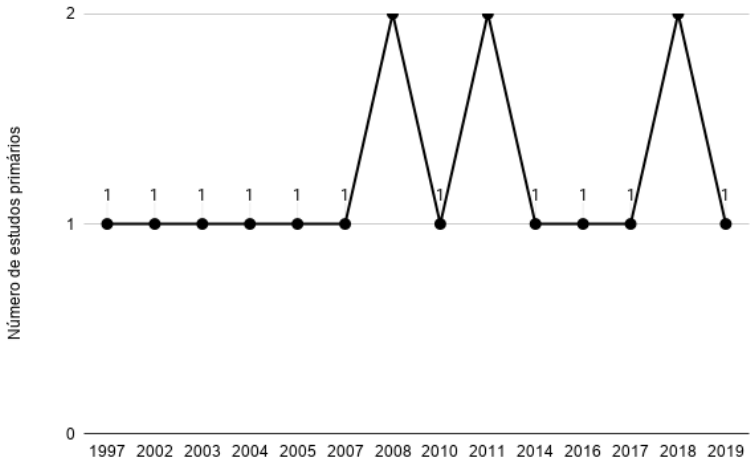

Figura 2. Visão geral dos estudos primários de acordo com o ano de publicação

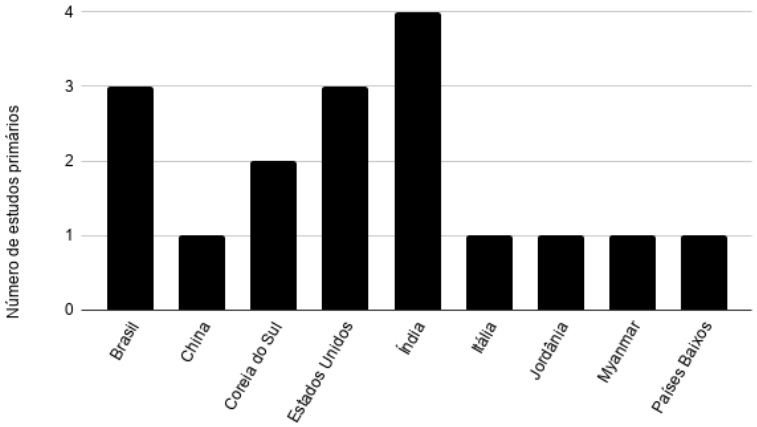

Figura 3. Visão geral dos estudos conforme o país de origem do primeiro autor

Segundo [Lientz and Swanson 1980], existem quatro tipos de manutenção de software: corretiva, adaptativa, perfectiva e preventiva. Observou-se que, referente ao tipo de manutenção a qual seriam empregadas métricas, métodos e ferramentas, cinco estudos não especificaram o tipo de manutenção. Três estudos aplicaram estimativa de esforço relacionado à manutenção corretiva, um estudo focou em manutenção adaptativa, dois estudos focaram em manutenção corretiva e adaptativa, dois focaram em manutenção corretiva e evolutiva e quatro estudos focaram em três tipos de manutenção: corretiva, adaptativa e evolutiva. Conforme ilustrado na Figura 4, pode-se observar a quantidade de estudos considerando separadamente cada tipo de manutenção.

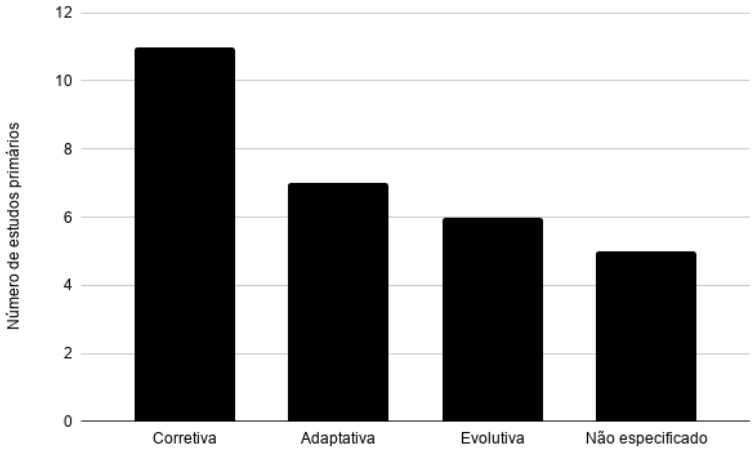

Figura 4. Visão geral dos estudos de acordo com o tipo de manutenção

Para responder a QP, na Tabela 1 são apresentados os métodos utilizados para estimativa de esforço encontrados no MS, juntamente com as categorias nas quais os artigos foram classificados. Observa-se que muitos dos artigos utilizaram métricas (11 de 17) tanto para gerar a estimativa (C_1), quanto para auxiliar outros métodos a gerarem a estimativa (C_1 e C_2). Dos 17 artigos, 13 apresentaram algum método de estimativa de esforço, porém apenas cinco apresentaram apenas o método sem utilizar de métricas ou uma nova ferramenta (C_2). Analisando a categoria C_3, apenas um estudo apresentou um *framework* em conjunto com métodos de estimativa de esforço (C_2 e C_3).

Tendo em vista os estudos que utilizam métricas para o cálculo da estimativa de esforço (categoria 1), observa-se que a métrica mais utilizada é o número de linhas de código (6 de 11 estudos). No entanto, os estudos utilizaram cálculos diferentes dessa métrica: alguns mediram o tamanho do sistema para complementar as demais métricas, outros exploravam a quantidade de linhas alteradas, inseridas ou excluídas para gerar a estimativa. Dos demais estudos, quatro utilizaram pontos de função e apenas um estudo [Chandra et al. 2017] aplicou métricas de orientação a objetos, como número de métodos por classe e profundidade de herança. A Tabela 2 mostra as métricas utilizadas para o cálculo de estimativa de esforço para cada estudo baseado em métricas (C_1).

Quanto à categoria 2, ou seja, técnicas e abordagens para o cálculo da estimativa, observa-se que o método mais usado foi a análise de regressão linear (5 de 13 estudos), o qual permite chegar a um valor estimado a partir de outros valores analisados de uma base de dados históricos. Além disso, quatro estudos utilizaram rede neural, o que acabou

Tabela 1. Visão geral dos estudos primários e seus métodos utilizados

Artigo	**Categoria**	**Métodos**
[Niessink and Van Vliet 1997]	C1, C2	Pontos de Função e Analogia
[Leung 2002]	C2	Analogia com o método vizinho virtual (AVN)
[Ahn et al. 2003]	C1	Pontos de Função
[Hayes et al. 2004]	C1	Modelo adaptável baseado em métricas
[De Lucia et al. 2005]	C1, C2	Regressão linear multivariada
[Song et al. 2007]	C1, C2	Modelo probabilístico baseado em rede Bayesiana
[Shukla and Misra 2008]	C1, C2	Rede Neural
[Tenório Jr et al. 2008]	C1, C2	Regressão Linear pelo Menor Quadrado (LSLR) e Pontos de Função
[Thaw et al. 2010]	C2	Mapa auto-organizado em Rede Neural
[Bharathi and Shastry 2011]	C2	Rede Neural
[Nguyen et al. 2011]	C1	COCOMO
[Alomari et al. 2014]	C1, C2	Análise de regressão e fatiamento do código
[Miguel et al. 2016]	C2, C3	Similaridade textual, reputação dos desenvolvedores e análise de regressão
[Chandra et al. 2017]	C1, C2	Análise de regressão e Support Vector Machine (Univariada e Multivariada)
[Hira and Boehm 2018]	C1	COSMIC Pontos de Função com SNAP
[Lélis et al. 2018]	C2	Reputação dos desenvolvedores
[Pillai and Madhukumar 2019]	C2	Julgamento de especialista

Tabela 2. Visão geral dos estudos primários e suas métricas utilizadas

Artigo	**Métricas**
[Niessink and Van Vliet 1997]	Pontos de Função
[Ahn et al. 2003]	Pontos de Função
[Hayes et al. 2004]	Linhas e operadores alterados
[De Lucia et al. 2005]	Tamanho do sistema (KLOC), número de tarefas
[Song et al. 2007]	Experiência do mantenedor, tamanho do software, características estruturais
[Shukla and Misra 2008]	Complexidade, número de linhas, número de arquivos
[Tenório Jr et al. 2008]	Pontos de Função
[Nguyen et al. 2011]	Linhas inseridas, modificadas e excluídas
[Alomari et al. 2014]	Tamanho total do sistema, tempo de atraso, intervalo de tempo entre versões, número de *hashes* modificado
[Chandra et al. 2017]	Número de métodos por classe, acoplamento, falta de coesão, número de filhos, profundidade da herança
[Hira and Boehm 2018]	Pontos de Função

se tornando o segundo método mais aplicado. Dos demais métodos podemos citar: julgamento de especialista na atividade, similaridade textual e analogia que fazem a busca de atividades já realizadas para assim gerar a estimativa de esforço, dentre outros métodos apresentados na Tabela 1.

Ao analisar a categoria 3, referente a ferramentas e *frameworks*, obteve-se apenas um estudo que se enquadrou nesta categoria [Miguel et al. 2016]. O estudo apresentou um *framework* que além de gerar as estimativas segundo diversas abordagens, apresenta também seis tipos de visualizações em formato de diagramas para ajudar no acompanhamento das atividades de manutenção. Dentre as visualizações, se destacam a visualização da reputação dos desenvolvedores ao longo do tempo, e uma representação em regressão linear do esforço real utilizado nas atividades de manutenção.

4. Ameaças à Validade

A *validade interna* é relacionada ao tratamento dos resultados obtidos. Durante a seleção dos artigos, o fator humano pode ter tendenciado a pesquisa e, consequentemente, afetado os resultados do MS. Para mitigar esta ameaça, toda a condução do MS foi realizada em pares (primeiro e segundo autor) e documentada via planilha compartilhada, a qual foi inspecionada posteriormente por um revisor (terceiro autor).

A *validade externa* é relacionada às chances da generalização dos estudos obtidos. Este MS extraiu estudos de quatro grandes bases de busca a partir de uma *string* de busca abrangente, com apenas dois termos. Em consequência dessa busca ampla, uma grande quantidade de estudos foram retornados no início da busca. No entanto, não foi utilizado um estudo de controle. Dessa forma, há a possibilidade de existirem outros estudos relevantes e que não foram incluídos no MS.

5. Considerações Finais

Neste trabalho foi realizado um mapeamento sistemático a partir das diretrizes propostas por [Kitchenham 2004], com o principal objetivo de encontrar quais métodos, métricas ou ferramentas estavam sendo utilizadas na manutenção de software para realizar estimativa de esforço de atividades. Deste modo, foram apresentados e discutidos os 17 estudos selecionados do mapeamento, os quais foram divididos em três categorias: C_1 – Métricas, C_2 – Métodos e C_3 – Ferramentas.

Os resultados apontam que os métodos são mais utilizados que as métricas, apesar dos dois poderem ser usados em conjunto. Podemos citar que a métrica mais utilizada foi linhas de código, podendo ser usada para medir o tamanho total do sistema, ou identificar linhas alteradas, incluídas ou excluídas. Já o método mais utilizado foi a análise de regressão linear, o que representa que muitos estudos preferem utilizar dados históricos para gerar novas estimativas. Quanto às ferramentas, obteve-se apenas um *framework* proposto, composto por seis tipos de visualizações para facilitar para o usuário o acompanhamento das atividades.

Entre os estudos é possível analisar que poucos levaram em consideração a opinião dos especialistas no projeto quando se trata de estimativas, os quais podem fornecer dados baseados em suas experiências. Diante dos estudos resultantes do MS, observa-se que todos apresentaram um método que aprimora a estimativa, porém apenas o ar-

tigo [Miguel et al. 2016] apresentou uma maneira de ajudar o usuário a gerar essas estimativas. Assim, uma lacuna de pesquisa seria o desenvolvimento de *frameworks* ou ferramentas que facilitassem ao usuário a geração das novas estimativas de esforço para manutenção de software.

Portanto, a principal contribuição deste trabalho é a obtenção de uma visão geral das pesquisas que estavam sendo realizadas de estimativa de esforço e que focassem nas atividades de manutenção de software. Em vista disso, como trabalho futuro, pretende-se realizar um survey, o qual teria como objetivo identificar o que os desenvolvedores levam em consideração no momento de estimar uma nova atividade e quais métodos estão sendo utilizados na indústria. Desse modo, visa-se aprimorar o método de geração de estimativa e deixá-la mais próxima do contexto atual do projeto.

Referências

Ahn, Y., Suh, J., Kim, S., and Kim, H. (2003). The software maintenance project effort estimation model based on function points. *Software maintenance and evolution: Research and practice*, 15(2):71–85.

Alomari, H. W., Collard, M. L., and Maletic, J. I. (2014). A slice-based estimation approach for maintenance effort. In *30th International Conference on Software Maintenance and Evolution*, pages 81–90.

Bharathi, V. and Shastry, U. (2011). Neural network based effort prediction model for maintenance projects. In *11th International Conference on Software Process Improvement and Capability Determination*, pages 236–239.

Chandra, D., Choudhary, M., and Gupta, D. (2017). Prophecy of software maintenance effort with univariate and multivariate approach: By using support vector machine learning techniquewith radial basis kernel function. In *6th International Conference on Computing, Communication and Automation*, pages 876–880.

De Lucia, A., Pompella, E., and Stefanucci, S. (2005). Assessing effort estimation models for corrective maintenance through empirical studies. *Information and Software Technology*, 47(1):3–15.

Erlikh, L. (2000). Leveraging legacy system dollars for e-business. *IT Professional*, 2(3):17–23.

Hayes, J. H., Patel, S. C., and Zhao, L. (2004). A metrics-based software maintenance effort model. In *8th European Conference on Software Maintenance and Reengineering*, pages 254–258.

Hira, A. and Boehm, B. (2018). COSMIC function points evaluation for software maintenance. In *11th Innovations in Software Engineering Conference*, page 4.

Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, 33(2004):1–26.

Lélis, C. A., Miguel, M. A., Araújo, M. A. P., David, J. M. N., and Braga, R. (2018). AD-reputation: a reputation-based approach to support effort estimation. In *Information Technology – New Generations*, pages 621–626. Springer.

Leung, H. K. (2002). Estimating maintenance effort by analogy. *Empirical Software Engineering*, 7(2):157–175.

Lientz, B. and Swanson, E. (1980). *Software maintenance management: a study of the maintenance of computer application software in 487 data processing organizations*. Addison-Wesley.

Miguel, M. A., Araujo, M. A. P., David, J. M. N., and Braga, R. (2016). A framework to support effort estimation on software maintenance and evolution activities. In *12th Brazilian Symposium on Information Systems: Information Systems in the Cloud Computing Era*, pages 232–239.

Nguyen, V., Boehm, B., and Danphitsanuphan, P. (2011). A controlled experiment in assessing and estimating software maintenance tasks. *Information and Software Technology*, 53(6):682–691.

Niessink, F. and Van Vliet, H. (1997). Predicting maintenance effort with function points. In *13th International Conference on Software Maintenance*, pages 32–39.

Pillai, S. and Madhukumar, R. (2019). An experiment to improve expert judgment software estimation through work breakdown structure. *Innovative Technology and Exploring Engineering*, 8(7):2278–3075.

Shukla, R. and Misra, A. K. (2008). Estimating software maintenance effort: a neural network approach. In *1st India Software Engineering Conference*, pages 107–112.

Song, T.-H., Yoon, K.-A., and Bae, D.-H. (2007). An approach to probabilistic effort estimation for military avionics software maintenance by considering structural characteristics. In *14th Asia-Pacific Software Engineering Conference*, pages 406–413.

Tenório Jr, N. N., Ribeiro, M. B., and Ruiz, D. D. (2008). A quasi-experiment for effort and defect estimation using least square linear regression and function points. In *32nd Annual IEEE Software Engineering Workshop*, pages 143–151.

Thaw, T., Aung, M. P., Wah, N. L., Nyein, S. S., Phyo, Z. L., and Htun, K. Z. (2010). Comparison for the accuracy of defect fix effort estimation. In *2nd International Conference on Computer Engineering and Technology*, pages 550–554.

Vazquez, C. E., Simões, G. S., and Albert, R. M. (2013). *Análise de Pontos de Função: Medição, estimativas e gerenciamento de projetos de software*. Érica, São Paulo.