Métodos e Ferramentas para Auxílio ao Ensino de Engenharia de Software: uma Revisão Sistemática

Douglas Lusa Krug[a,*], Jean M. Simão[b,*], Laudelino Cordeiro Bastos[b,*]

[a] *Instituto Federal do Paraná - IFPR - União da Vitória - PR*
[b] *Universidade Tecnológica Federal do Paraná - UTFPR - Curitiba - PR*

Abstract

A Engenharia de Software é uma área de conhecimento da Ciência da Computação que estuda a construção de sistemas de forma eficaz e eficiente para atender as exigências dos clientes. O campo de estudo desta área é grande, passando pelo levantamento de requisitos, desenho e construção do sistema, planejamento e gerenciamento dos processos. A pluralidade de subáreas pertencentes à Engenharia de Software torna o ensino desta área complexo, sendo necessário muitas vezes métodos e ferramentas que auxiliam o docente no processo de ensino-aprendizagem, principalmente quanto a aplicação prática. Este artigo apresenta uma revisão sistemática da literatura a respeito de métodos e ferramentas que auxiliam no ensino da Engenharia de Software, e tem por objetivo mapear as áreas onde estes métodos são aplicados, verificando a interferência no processo de ensino-aprendizagem e identificando possíveis campos para aprimoramento.

Keywords: ‘‘Software Engineering Education Methods”, “Software Engineering Education Tools”, “Software Engineering Education”

*Corresponding author
Email addresses: douglas.krug@ifpr.edu.br (Douglas Lusa Krug), jeansimao@utfpr.edu.br (Jean M. Simão), bastos@dainf.ct.utfpr.edu.br (Laudelino Cordeiro Bastos)

1. Introdução

Engenharia de Software (ES) segundo a definição do SWEBOK [1] é a aplicação de uma sistemática e disciplinada abordagem quantificável para o desenvolvimento, operação e manutenção de software.

Seguindo a mesma linha o documento criado em conjunto pela Association for Computing Machineri (ACM) e pela IEEE Computer Society, que serve como guia para currículos de cursos de Ciências da Computação [2], define ES como disciplina preocupada com a aplicação da teoria, conhecimento e prática para construção eficaz e eficiente de sistemas de software confiáveis que satisfazem as exigências dos clientes e usuários.

A área de conhecimento de ES é de extrema importância para formação de futuros profissionais para o mercado de software, que podem atuar em diversas funções, desde programadores até gerentes de projeto.

O currículo de um curso da área de CS, seja em nível técnico profissionalizante ou superior é composto por diferentes disciplinas, algumas delas ligadas diretamente com a área de ES [2].

Algumas destas disciplinas podem ser consideradas como disciplinas básicas da área, como por exemplo, Introdução à Algoritmos e Lógica de Programação. Outras avançadas, como Programação Avançada e Análise Orientada a Objetos.

Ainda existem currículos que contam com disciplinas específicas de ES, que tratam diretamente os métodos desta área de conhecimento.

No documento elaborado pela Sociedade Brasileira da Computação (SBC) [3], podemos encontrar as disciplinas base, avançadas e específicas da área de ES, demonstrando a gama de disciplinas em que o tema pode ser trabalhado.

Independente do nível de ensino, alguns conteúdos da área se tornam complexos e abstratos para os estudantes, exigindo alternativas de ensino que devem ser utilizadas pelo docente.

O desenvolvimento da educação em ES é um processo complexo e iterativo pois a disciplina e o ambiente estão sendo modificados ao longo do tempo e necessitam de ajustes [4].

A área de ES é de difícil aprendizagem para os alunos de diversos níveis de formação, devido à ampla quantidade de conhecimentos envolvidos. Geralmente, alunos de cursos de CS compreendem a resolução de problemas básicos, porém enfrentam dificuldades quando são expostos para problemas complexos, os quais serão encontrados diariamente no mercado de trabalho.

O método de ensino para as disciplinas de ES deve ser capaz de simular os problemas que os alunos irão enfrentar na sua vida profissional, além de simular o ambiente em que encontrarão estes problemas. Este é um desafio para os docentes das disciplinas da área, que devem, além de seu conhecimento buscar apoio em métodos e ferramentas para o ensino destas disciplinas.

Conforme Koch e Landes [4], para identificar abordagens didáticas que são apropriados para fomentar competências dos alunos em um determinado contexto, novos métodos de ensino precisam ser experimentados e testados, e os resultados precisam ser avaliados.

O presente artigo relata uma Revisão Sistemática da Literatura (RSL) sobre métodos e ferramentas para o ensino de ES, buscando levantar informações para elaborar o estado da arte para a dissertação de mestrado, além de possíveis lacunas para serem trabalhadas.

Conforme Kitchenham e Charters [5] uma revisão sistemática da literatura significa avaliar e interpretar toda pesquisa relevante para uma questão em particular, um tópico, ou um fenômeno de interesse. A RSL visa apresentar uma justa evolução de um tópico de pesquisa, usando um método confiável, rigoroso e auditável.

1.1. Objeto de Estudo

Os objetos de estudo para este artigo são os métodos e ferramentas que podem ser utilizadas para o ensino de ES, no contexto geral do conhecimento desta área.

1.2. Objetivos

A presente RSL visa atender aos objetivos listados a seguir:

Objetivo geral:.

- Coletar os métodos e ferramentas para auxílio ao ensino de ES, identificando a eficácia e espaço para aprimoramento.

Objetivos específicos:.

- Categorizar as áreas de ES apresentadas nos trabalhos selecionados para análise;
- Validar se os trabalhos analisados relatam interferência positiva quanto à utilização das ferramentas utilizadas no ensino de ES;
- Identificar campo aberto para desenvolvimento e/ou aprimoramento do método ou ferramenta para auxílio ao ensino de ES.

2. Método

Esta RSL é baseada no guia escrito por Kitchenham e Charters [5], e é composta pelas fases de planejamento, realização da revisão, e o relato da revisão.

O objetivo de utilizar uma RSL neste trabalho é identificar possíveis lacunas na área de pesquisa, assim sugerindo áreas para maiores investigações.

2.1. Questão de Pesquisa

Para realizar a RSL, que servirá como base para o estado da arte do trabalho de dissertação de mestrado, as seguintes questões de pesquisa foram consideradas:

Q1. A utilização dos métodos e ferramentas disponíveis para auxílio ao ensino de Engenharia de Software apresentaram interferência positiva?

Q2. Quais subáreas da ES estão sendo atendidas pelos métodos e ferramentas encontrados?

Q3. Os métodos encontrados podem ser aprimorados?

2.2. Estratégia de Pesquisa

Uma RSL deve levar em conta os trabalhos primários realizados sobre o tema escolhido, visando responder as questões de pesquisa.

Para isso, uma estratégia de pesquisa deve ser escolhida. A pesquisa deve ser concisa, e estar documentada para que possa ser replicada no futuro.

Para esta RSL as bases de artigos científicos escolhidas foram as listadas na tabela 1.

Tabela 1: Bases de Artigos

Base	Data de Consulta
IEEE Xplore	15/07/2016
Scopus	15/07/2016
ACM Digital Library	15/07/2016
Science Direct	15/07/2016

De acordo com a definição das questões de pesquisa, os seguintes termos foram escolhidos para extrair os artigos das bases mencionadas anteriormente: "Software Engineering Education Methods"; "Software Engineering Education Tools"; "Software Engineering Education System". Desta forma o termo de pesquisa utilizado foi: ("Software Engineering Education"AND (Methods OR Tools OR System)).

A extração dos artigos foi realizada considerando apenas artigos publicados a partir do ano de 2005.

2.3. Critérios de Inclusão e Exclusão

Através dos resultados obtidos a partir da pesquisa nas bases de artigos, foi realizada uma triagem dos artigos, seguindo critérios de inclusão e exclusão.

CI1. Foram incluídos artigos que apresentaram métodos ou ferramentas para auxílio ao ensino de Engenharia de Software.

CI2. Foram incluídos artigos que demonstraram os resultados através de aplicação do método ou ferramenta.

CI3. Foram incluídos artigos que estavam completamente no idioma Inglês.

CE1. Foram excluídos artigos que não estavam completamente disponíveis.

CE2. Foram excluídos artigos que não demonstraram aplicação de métodos ou ferramentas para auxílio ao ensino de Engenharia de Software.

Após a busca dos artigos utilizando os termos de pesquisa mencionados anteriormente, 1244 artigos foram encontrados. Entre eles, foram encontrados 297 artigos em duplicidade, após corrigir as duplicidades restaram 947 artigos.

Inicialmente os critérios de inclusão e exclusão foram aplicados ao título e ao resumo, através desta triagem restaram 87 artigos.

Para os 87 artigos que foram selecionados, foi realizada uma análise do documento completo, aplicando também os critérios de inclusão e exclusão. Após esta etapa 28 artigos foram selecionados para a RSL. Os detalhes estão apresentados na figura 1.

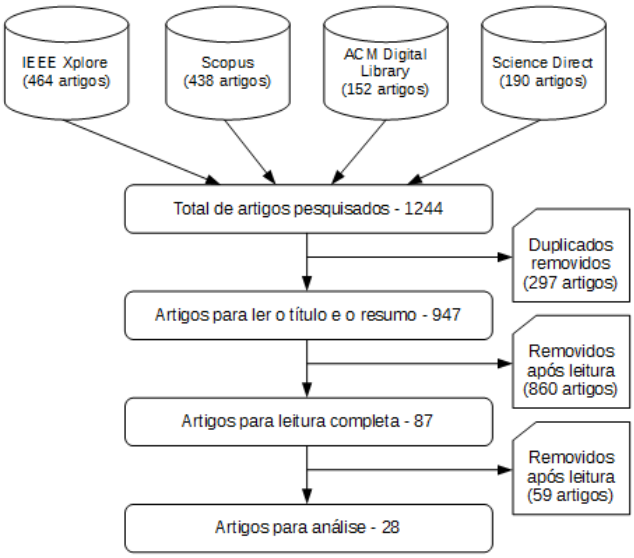

Figura 1: Triagem de Artigos

A figura 2 demonstra os artigos selecionados quantificando de acordo com o ano de publicação.

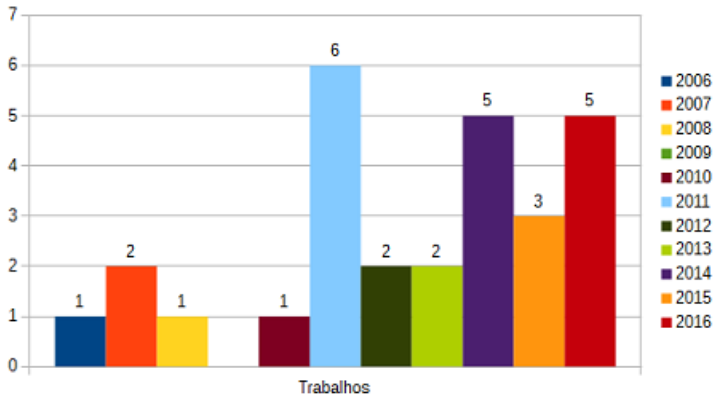

Figura 2: Artigos por ano de publicação

3. Análise

3.1. Extração de Dados

A fim de responder as questões de pesquisa deste trabalho, os artigos listados na tabela 2, foram analisados e algumas informações foram tabuladas conforme relatado nas seções a seguir.

Tabela 2: Trabalhos Selecionados

Trabalho	Ano	Método/Ferramenta
[6]	2012	*Role-playing game*
[7]	2011	*Game-based learning*
[8]	2011	*Project-based learning*
[9]	2016	*Instructional Unit*
[10]	2016	*Project-based learning*
[11]	2016	*Integrated development environment*
[12]	2007	3-D *Online Virtual World*
[13]	2011	*Game-based learning*
[14]	2015	*Serious game*
[15]	2011	*Problem-based Learning*
[16]	2014	*Industry-oriented teaching techniques*
[17]	2015	*Custom eXtreme Programing*
[18]	2012	*Project-based learning + Problem-based learning*
[19]	2016	*Agile Coaching*
[20]	2015	*Active and Cooperative Learning*
[21]	2014	*Framework for System Development*
[22]	2013	*Manual paper and pencil game*
[23]	2008	*Education Toolkit (Project and Tools)*
[24]	2011	*Test-driven learning (webIDE)*
[25]	2014	*Scrum simulation game*
[26]	2013	*Digital CRC Sessions (CREWSpace)*
[27]	2014	*Role playing + realistic scenario*
[28]	2011	*Simulation Engine (AMEISE)*
[29]	2014	*LEGO-based Scrum simulation game*
[30]	2016	*Automated Validation of UML Class Diagram (UMLtest)*
[31]	2007	*Game-based learning*
[32]	2010	*Complex problems with LEGO MINDSTORMS NXT*
[33]	2006	*Writing as a tool for learning*

Para responder a Q1, a tabela 3 demonstra os trabalhos informando se houve interferência positiva, neutra, ou negativa no aprendizado através da utilização dos métodos e ferramentas conforme validação realizada em cada artigo.

Tabela 3: Interferência no Aprendizado

Interferência	Trabalhos
Positiva	[6][9][10][11][12][13][14][15][16][17][18][19][20] [21][22][23][25][26][27][28][29][30][31][32][33]
Neutra	[7][8][10][24]
Negativa	

A Q2 visa identificar quais subáreas da ES estão sendo atendidas pelos métodos e ferramentas relatados nos artigos analisados, esta relação está representada na tabela 4. Os trabalhos [10] e [26] atendem a 2 subáreas.

A definição das subáreas foi baseada no SWEBOK [1].

Tabela 4: Subáreas

Subárea	Trabalhos
Requisitos de Software	[10][13][16]
Desenho de Software	[8][11][18][20][23][26][27][33]
Construção de Software	[21][24][26][30]
Engenharia de Software	[6][12][15][17][19][32]
Gerenciamento de Engenharia de Software	[9][22][25][28][29][31]
Processos de Engenharia de Software	[7][10][14]

3.2. Discussão

Através da RSL realizada, é possível observar que em 25 trabalhos é relatada interferência positiva do método ou ferramenta como auxílio ao ensino das disciplinas de ES.

Em apenas 4 trabalhos a conclusão dos autores é que não foi possível observar interferência positiva nem interferência negativa, ou seja a interferência é neutra. Em alguns destes casos os dados obtidos não foram estatisticamente significantes. O trabalho [10] está citado com interferência positiva e neutra, pois o mesmo atende áreas distintas da ES, sendo que apresentou interferência positiva na área de Requisitos de Software e neutra na área de Processos de Engenharia de Software.

É importante salientar que em nenhum dos trabalhos analisados foi apresentada interferência negativa no processo de ensino-aprendizagem.

No ponto de vista do autor, e com base no que está relatado na tabela 3, a resposta para a Q1-"A utilização dos métodos e ferramentas disponíveis para auxílio ao ensino de Engenharia de Software apresentaram interferência positiva?"é sim, de acordo com os trabalhos analisados a utilização de métodos e ferramentas para o apoio ao ensino de ES demonstra interferência positiva ao processo de ensino-aprendizagem.

A partir da análise dos artigos, os mesmos foram separados em subáreas da ES, seguindo as definições contidas no SWEBOK [1].

Alguns trabalhos não especificaram uma área seguindo a classificação do SWEBOK, estes trabalhos foram classificados como uma área geral - Engenharia de Software.

Os 28 trabalhos foram classificados nas seguintes áreas: Requisitos de Software; Desenho de Software; Construção de Software; Engenharia de Software; Gerenciamento de Engenharia de Software; e Processos de Engenharia de Software.

A lista destas 6 áreas responde à pergunta Q2-"Quais subáreas da ES estão sendo atendidas pelos métodos e ferramentas encontrados?".

Através desta resposta, demonstrada na tabela 4, é possível observar que a utilização de métodos e ferramentas para apoio ao ensino é utilizado em diversas áreas de ES, onde o ensino não se atém somente a aulas expositivas.

A respeito da Q3-"Os métodos encontrados podem ser aprimorados?", todos os trabalhos analisados deixam espaço para novas aplicações e aprimoramento.

Embora diversas áreas da ES estejam sendo trabalhadas, ainda existem lacunas quanto à utilização de métodos e ferramentas para auxílio ao ensino de ES, principalmente nas áreas de Desenho de Software e Construção de Software.

4. Conclusão

Esta revisão sistemática da literatura foi escrita com o objetivo de verificar se os métodos e ferramentas utilizados como auxílio ao ensino das disciplinas da área de Engenharia de Software demonstram interferência positiva no processo de ensino-aprendizagem.

De acordo com o levantamento realizado, foram selecionados 28 trabalhos que apresentam métodos ou ferramentas de auxílio ao ensino de ES, sendo estes criados pelos autores ou apenas relatos da aplicação de método já existente. Todos os trabalhos tiveram sua aplicação avaliada.

Foi possível concluir que os métodos e ferramentas apresentaram interferência positiva no processo de ensino-aprendizagem, reforçando a ideia de que disciplinas da área de ES necessitam de complemento às aulas expositivas.

Os trabalhos foram classificados em subáreas da ES, conforme tabela 4, através desta classificação foi possível notar que os métodos e ferramentas não se atém apenas à algumas subáreas, mas englobam diversas subáreas da ES.

Ainda é possível identificar que todos os trabalhos deixam espaço para aperfeiçoamento e novas experiências de aplicação.

References

[1] P. Bourque, R. E. Fairley (Eds.), SWEBOK: Guide to the Software Engineering Body of Knowledge, version 3.0 Edition, IEEE Computer Society, Los Alamitos, CA, 2014.
URL `http://www.swebok.org/`

[2] A. f. C. M. A. Joint Task Force on Computing Curricula, I. C. Society, Computer Science Curricula 2013: Curriculum Guidelines for Undergradu-

ate Degree Programs in Computer Science, ACM, New York, NY, USA, 2013, 999133.

[3] S. B. de Computação, Currículo de referência da sbc para cursos de graduação em bacharelado em ciência da computação e engenharia de computação (2005).

[4] M. Koch, D. Landes, A recommender system for didactical approaches in software engineering education, Vol. 1227, 2014, pp. 140–143, cited By 1.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84921879742&partnerID=40&md5=543f9bd9c7b3eaf20e06eb7b63475fb8

[5] B. A. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering (2007).

[6] S. Zuppiroli, P. Ciancarini, M. Gabbrielli, A role-playing game for a software engineering lab: Developing a product line, in: 2012 IEEE 25th Conference on Software Engineering Education and Training, 2012, pp. 13–22. doi:10.1109/CSEET.2012.39.

[7] T. Lynch, M. Herold, J. Bolinger, S. Deshpande, T. b. Bihari, J. Ramanathan, R. Ramnath, An agile boot camp: Using a lego®-based active game to ground agile development principles, 2011, cited By 0. doi:10.1109/FIE.2011.6142849.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84858164710&partnerID=40&md5=0d57770d64b88fdfd6e4f02e0af73180

[8] Z. Jeremic, J. Jovanovic, D. Gasevic, An environment for project-based collaborative learning of software design patterns, International Journal of Engineering Education 27 (1 PART 1) (2011) 41–51, cited By 8.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-79958847658&partnerID=40&md5=a38b2018c8be90caa7ea05704da3b118

[9] R. Q. Gonçalves, C. G. V. Wangenheim, An instructional unit for teaching project management tools aligned with pmbok, in: 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET), 2016, pp. 46–55. doi:10.1109/CSEET.2016.10.

[10] G. Marsicano, F. F. Mendes, M. V. Fernandes, S. A. A. d. Freitas, An integrated approach to the requirements engineering and process modelling teaching, in: 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET), 2016, pp. 166–174. doi:10.1109/CSEET.2016.23.

[11] D. Kung, J. Lei, An object-oriented analysis and design environment, in: 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET), 2016, pp. 91–100. doi:10.1109/CSEET.2016.20.

[12] E. Ye, C. Liu, J. A. Polack-Wahl, Enhancing software engineering education using teaching aids in 3-d online virtual worlds, in: 2007 37th Annual Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports, 2007, pp. T1E–8–T1E–13. doi:10.1109/FIE.2007.4417884.

[13] T. Hainey, T. Connolly, M. Stansfield, E. Boyle, Evaluation of a game to teach requirements collection and analysis in software engineering at tertiary education level, Computers and Education 56 (1) (2011) 21–35, cited By 56. doi:10.1016/j.compedu.2010.09.008.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-77957990403&partnerID=40&md5=f6468913cfddd5e0b1c05d8b8f5f2b73

[14] R. O. Chaves, C. G. von Wangenheim, J. C. C. Furtado, S. R. B. Oliveira, A. Santos, E. L. Favero, Experimental evaluation of a serious game for teaching software process modeling, IEEE Transactions on Education 58 (4) (2015) 289–296. doi:10.1109/TE.2015.2411573.

[15] K. Tian, K. Cooper, K. Zhang, Improving software engineering education through enhanced practical experiences, in: Computer and Information Science (ICIS), 2011 IEEE/ACIS 10th International Conference on, 2011, pp. 292–297. doi:10.1109/ICIS.2011.53.

[16] M. Daun, A. Salmon, B. Tenbergen, T. Weyer, K. Pohl, Industrial case studies in graduate requirements engineering courses: The impact on student motivation, in: 2014 IEEE 27th Conference on Software Engineering Education and Training (CSEE T), 2014, pp. 3–12. doi:10.1109/CSEET.2014.6816775.

[17] J. J. Y. Chen, M. M. Z. Wu, Integrating extreme programming with software engineering education, in: Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2015 38th International Convention on, 2015, pp. 577–582. doi:10.1109/MIPRO.2015.7160338.

[18] Y. Zhang, Y. Liu, Management enhanced double pbl based reform in advanced programming design course, in: High Performance Computing and Communication 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS), 2012 IEEE 14th International Conference on, 2012, pp. 1658–1663. doi:10.1109/HPCC.2012.244.

[19] G. Rodríguez, . Soria, M. Campo, Measuring the impact of agile coaching on students' performance, IEEE Transactions on Education PP (99) (2016) 1–1. doi:10.1109/TE.2015.2506624.

[20] J. A. Pow-Sang, Replacing a traditional lecture class with a jigsaw class to teach analysis class diagrams, in: Interactive Collaborative Learning (ICL), 2015 International Conference on, 2015, pp. 389–392. doi:10.1109/ICL.2015.7318059.

[21] S. Zschaler, B. Demuth, L. Schmitz, Salespoint: A java framework for teaching object-oriented software development, Science of Computer Programming 79 (2014) 189 – 203, experimental Software and

Toolkits (EST 4): A special issue of the Workshop on Academic Software Development Tools and Techniques (WASDeTT-3 2010). doi:http://dx.doi.org/10.1016/j.scico.2012.04.005.
URL http://www.sciencedirect.com/science/article/pii/S016764231200069X

[22] C. G. von Wangenheim, R. Savi, A. F. Borgatto, Scrumia—an educational game for teaching {SCRUM} in computing courses, Journal of Systems and Software 86 (10) (2013) 2675 – 2687. doi:http://dx.doi.org/10.1016/j.jss.2013.05.030.
URL http://www.sciencedirect.com/science/article/pii/S0164121213001295

[23] D. Kim, S. Kim, S. Kim, S. Park, Software engineering education toolkit for embedded software architecture design methodology using robotic systems, in: 2008 15th Asia-Pacific Software Engineering Conference, 2008, pp. 317–324. doi:10.1109/APSEC.2008.58.

[24] T. Dvornik, D. S. Janzen, J. Clements, O. Dekhtyar, Supporting introductory test-driven labs with webide, in: 2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE T), 2011, pp. 51–60. doi:10.1109/CSEET.2011.5876137.

[25] M. Kropp, A. Meier, M. Mateescu, C. Zahn, Teaching and learning agile collaboration, in: 2014 IEEE 27th Conference on Software Engineering Education and Training (CSEE T), 2014, pp. 139–148. doi:10.1109/CSEET.2014.6816791.

[26] R. Lutz, S. Schäfer, S. Diehl, Teaching object-orientation with smartphones as digital crc cards, in: 2013 26th International Conference on Software Engineering Education and Training (CSEE T), 2013, pp. 89–98. doi:10.1109/CSEET.2013.6595240.

[27] G. Kotonya, J. Lee, Teaching reuse-driven software engineering through innovative role playing, 2014, pp. 276–282, cited By 2.

doi:10.1145/2591062.2591166.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84903589897&partnerID=40&md5=fe9e9ff40c25cabb963c956162e4deea

[28] A. Bollin, E. Hochmüller, R. T. Mittermeir, Teaching software project management using simulations, in: 2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE T), 2011, pp. 81–90. doi:10.1109/CSEET.2011.5876160.

[29] M. Paasivaara, V. Heikkilä, C. Lassenius, T. Toivola, Teaching students scrum using lego blocks, 2014, pp. 382–391, cited By 8. doi:10.1145/2591062.2591169.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-84903649666&partnerID=40&md5=ab72a287f81b9ccb1d4300ccf1c9a64d

[30] P. Herout, P. Brada, Uml-test application for automated validation of students' uml class diagram, in: 2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET), 2016, pp. 222–226. doi:10.1109/CSEET.2016.33.

[31] G. Taran, Using games in software engineering education to teach risk management, in: 20th Conference on Software Engineering Education Training (CSEET'07), 2007, pp. 211–220. doi:10.1109/CSEET.2007.54.

[32] M. W. Lew, T. B. Horton, M. S. Sherriff, Using lego mindstorms nxt and lejos in an advanced software engineering course, in: 2010 23rd IEEE Conference on Software Engineering Education and Training, 2010, pp. 121–128. doi:10.1109/CSEET.2010.31.

[33] A. I. Wang, C. F. Sorensen, Writing as a tool for learning software engineering, in: 19th Conference on Software Engineering Education Training (CSEET'06), 2006, pp. 35–42. doi:10.1109/CSEET.2006.46.

Apêndice

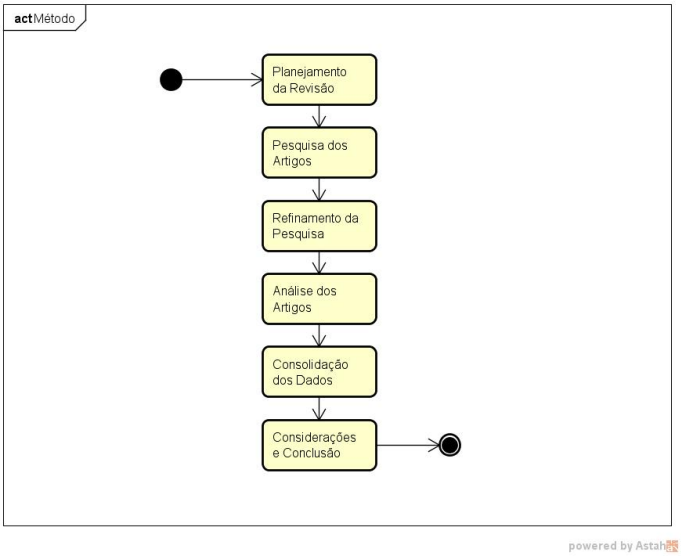

Figura 3: Diagrama de Atividades

Tabela 5: Cronograma de Atividades

Atividade	Período
Planejamento	09/06 - 30/06
Realização da Revisão	01/07 - 21/07
Extração de Dados	22/07 - 04/08
Síntese de Dados	05/08 - 13/08
Construção das Considerações	14/08 - 18/08
Conclusão do Artigo	19/08 - 01/09