Investigando Técnicas de Recomendação de Assets Híbridos de Software: Um Mapeamento Sistemático

[1]Rafael S. Torres, [1]Bruno Marcelo S. Ferreira, [1]Fábio P. Basso,
[1]Elder de Macedo Rodrigues, [1]Maicon Bernardino

[1]Universidade Federal do Pampa (UNIPAMPA), PPGES, Alegrete - RS - Brasil

torresrafa22gmail.com

Abstract. *Software reuse is an effective alternative in creating software with quality, for it allows the creation of new software from existing ones. Thanks to the vast availability of resources available to developers and to software houses, software reuse becomes more promising, yet more challenging. In order to make opportunistic reuse a more effective approach, machine learning techniques have been added to opportunistic reuse processes. This study summarizes a systematic mapping, characterizing how the machine learning area has supported the reuse of hybrid software assets. Our results indicate that there is no single approach that stands out from the others, opening space for comparative studies in the area.*

Resumo. *O reuso de software é uma alternativa efetiva para criar software com qualidade, por permitir criar novos produtos a partir de software já existente. Graças à vasta disponibilidade de recursos disponíveis para desenvolvedores e empresas, o reuso de software se torna mais promissor, e ao mesmo tempo, mais desafiador. De modo a tornar o reuso oportunista uma abordagem mais efetiva,técnicas da área de aprendizado de máquina vem sendo acrescentadas aos processos de reúso. Este estudo sumariza um mapeamento sistemático sobre como a área de aprendizado de máquina tem dado suporte ao reúso de assets híbridos de software. Nossos resultados apontam que não há uma única abordagem que se destaque das demais, abrindo espaço para estudos comparativos.*

1. Introdução

O reúso de software é tido como uma alternativa sólida para lidar com os problemas enfrentados no desenvolvimento de software, como falhas no produto e prazos limitados, já que permite criar novos produtos de software a partir de software já existente [Sametinger 1997], evitando parte do esforço e dos custos envolvidos no processo.

Atingir o objetivo de reusar com sucesso artefatos de software, porém, demanda planejamento e, dependendo da forma que é praticado o reúso, exige certo esforço por parte dos mantenedores e comsumidores dos *assets*. Para que esses esforços sejam minimizados e a eficiência do reúso seja intensificada, é necessário o suporte de técnicas/ferramentas para encontrar os artefatos corretos levando em consideração o problema a ser resolvido pelo desenvolvedor.

Neste artigo, relatamos os resultados obtidos a partir de um mapeamento sistemático que objetiva identificar como a área de aprendizado de máquina vem dando

suporte ao chamado reúso oportunista, caracterizado por buscas e recomendações de *assets* por meio de repositórios de reúso.

O restante deste trabalho está organizado como segue: A Seção 2 introduz os dois principais conceitos para o acompanhamento teórico deste trabalho. A Seção 3 apresenta o protocolo adotado na revisão de literatura e a Seção 4 discute os resultados do estudo executado. A Seção 5 apresenta a análise dos artigos para as questões de pesquisa investigadas. Por fim, a Seção 6 encerra com algumas considerações finais.

2. Embasamento

Data Mining pode ser definido como o processo de analisar conjuntos de dados e inferir padrões destes, através de algoritmos de aprendizado de máquina e métodos estatísticos. De acordo com [Fayyad et al. 1996], já era notória a necessidade de adotar técnicas de aprendizado de máquina a fim de automatizar a análise de dados, que antes era realizada manualmente, devido ao crescimento exponencial de informações disponíveis.

Ainda de acordo com [Fayyad et al. 1996], pode-se categorizar as diferentes abordagens para mineração de dados como: Classificação, Regressão, Aglomeração, Sumarização, Modelo de Dependência, e Detecção de anomalias.

Neste trabalho, buscou-se compreender qual é a relação entre data mining e reúso oportunista, que ocorre por meio de *assets* e repositórios. Um *asset*, no contexto da engenharia de software, pode ser caracterizado como qualquer artefato reutilizável produzido durante o ciclo de vida do desenvolvimento de software. De acordo com a definição dada pela OSLC [ams 2012], *asset* é qualquer coisa que pode ser mantida a fim de gerar valor. No contexto de software, *assets* podem ser código-fonte (funções, classes, ...), modelos, ferramentas e componentes. *Assets* híbridos por sua vez, representam artefatos de diferentes naturezas, um grupo heterogêneo de recursos. Podem se diferenciar por estarem em diferentes linguagens, ferramentas, abordagens, etc.

3. Mapeamento Sistemático

Seguindo as diretrizes de [Petersen et al. 2015], esta seção apresenta o protocolo de mapeamento sistemático utilizado. Para especificar o objetivo dessa pesquisa, foi utilizada a abordagem GQM [Caldiera and Rombach 1994], apresentada na Tabela 1

Tabela 1. Objetivo da pesquisa.

Objetivo	Com o objetivo de identificar e classificar
Questão	técnicas, metodologias, desafios, tendências
Objeto	mineração de dados de repositórios de *assets*, componentes e código
Viewpoint	do ponto de vista do pesquisador

Então, foram concebidas as seguintes questões de pesquisa:

- **Q1**: Quais são as propostas de técnicas/metodologias/guidelines propostas e aplicadas na área de mineração de dados úteis à aquisição/reúso de software/desenvolvimento?
- **Q2**: Que mecanismos de recomendação são utilizados?
- **Q3**: Quais são as tendências das pesquisas na área?

Tabela 2. Bases utilizadas na pesquisa.

Scopus	https://www.scopus.com/home.uri
IEEE	https://ieeexplore.ieee.org
Springer	https://www.springer.com
ACM	https://dl.acm.org/

A pesquisa foi realizada em 4 bases digitais de artigos amplamente conhecidas e utilizadas pela comunidade científica [Chen et al. 2010], apresentadas na Tabela 2.

A *string* de busca foi formulada através da estratégia PICo [Pregunta 2007], como mostra a Tabela 3. Vale ressaltar que foram necessárias algumas modificações específicas para algumas das bases pesquisadas, suprimidas neste artigo por motivos de espaço.

Tabela 3. Aplicação da estratégia PICo na formação da string de busca genérica

P	”data mining”OR ”big data”OR ”data extraction”OR “knowledge management” OR “knowledge discovery”
I	”asset management specification”OR ams OR ”reuse repository”OR ”component repository”OR ”mde repository”OR ”code recommender”OR ”recommender system”OR ”asset recommender”OR ”software asset”OR ”component asset”OR ”reusable asset”OR RAS
Co	“technology transfer” OR “software acquisition” OR “integration” OR “software engineering” OR “software development” OR “software reuse”

A seguir, apresenta-se os critérios de inclusão e de exclusão.

Critérios de Inclusão:

1. Artigos que apresentem técnicas/metodologias/guidelines para mineração de dados.
2. Artigos que relatam estudos de caso sobre técnicas/metodologias/guidelines para mineração de dados.
3. Artigos que tratem de sistemas de recomendação de *assets*/componentes/código em cenários de aquisição de software.

Critérios de Exclusão:

1. Estudos que não envolvam mineração de dados
2. Estudos que não estejam em inglês
3. Estudos duplicados
4. Estudos que não estejam disponíveis para download
5. Estudos que sofreram de retratação ou foram invalidados
6. Estudos que não são primários
7. Estudos que não tratem de transferência de tecnologia do domínio da engenharia de software nem de aquisição ou reúso de software.

4. Execução

Inicialmente, foram obtidos 888 artigos, como mostra a Tabela 4. A busca foi realizada considerando título, resumo e palavras-chave.

Após obter-se os resultados das bases, foram aplicados os critérios de exclusão. A Tabela 5 apresenta a quantidade de artigos desconsiderados.

Além disso, 55 artigos estavam repetidos entre as bases, sendo que destes, 44 estavam entre os descartados, e 11 entre os aceitos, resultando em 42 artigos aprovados após a leitura dos títulos, *abstract* e palavras-chave, e 21 após a leitura do texto completo.

Tabela 4. Artigos retornados

Base	Resultados
IEEE	78
ACM	294
Scopus	201
Springer	315
	888

Tabela 5. Resultado da aplicação dos critérios de exclusão.

Base \ Critério	CE1	CE6	CE7	Total
IEEE	4	4	61	69
ACM	14	7	262	281
Scopus	12	29	147	185
Springer	71	81	145	297
				832

A Figura 1 apresenta o processo descrito.Os artigos selecionados se encontram no Google Drive [1].

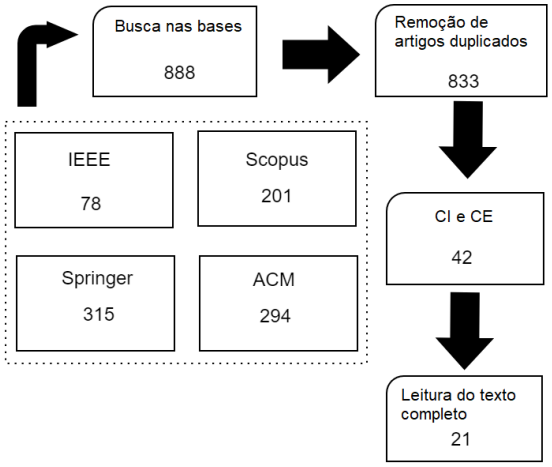

Figura 1. Processo de seleção de artigos.

5. Análise

5.1. Técnicas aplicadas (Q1)

Esta seção responde a **Q1**: Quais são as propostas de técnicas/metodologias/guidelines propostas e aplicadas na área de mineração de dados úteis à aquisição/reúso de software/desenvolvimento?

Pode-se observar dois focos diferentes em relação às propostas dos trabalhos obtidos: 1) utilizar técnicas de clusterização e organização de repositórios de *assets*; e 2) utilizar técnicas de filtragem e exploração de dados à fim de recomendar *assets* que possam ser úteis dependendo do contexto. Em um contexto de reúso oportunista, quando não há o reúso planejado, um repositório organizado daria suporte à busca manual por

[1] https://drive.google.com/drive/folders/1lTQPbAVb4qpij_MVuULYwZSh5THz-lBz?usp=sharing

um *asset*. Como apontado por [Wang and Ren 2011], um repositório organizado de maneira lógica é uma maneira efetiva de facilitar a busca de componentes. Um método de filtragem também poderia dar suporte a uma recomendação automatizada por parte do ambiente de desenvolvimento, baseando-se no contexto do desenvolvedor. Além disso, alguns trabalhos também buscam recursos em ambientes web. A relação de fases abordadas na recomendação está representada na Tabela 6.

Tabela 6. Fases identificadas nos trabalhos obtidos.

Artigo	Busca	Organização	Recomendação
[Ye and Lo 2000]		x	
[Nakkrasae and Sophatsathit 2004]		x	
[Li et al. 2004]			x
[Wang et al. 2004]		x	
[McCarey et al. 2005]			x
[Wu et al. 2007]		x	
[Bajracharya et al. 2009]	x	x	
[Martins et al. 2009]			x
[Wang and Ren 2011]		x	
[Dumitru et al. 2011]	x		x
[Heinemann 2012]			x
[Kumar et al. 2011]		x	
[Sayyad et al. 2012]			x
[Vodithala and Pabboju 2015]		x	
[Vescan 2015]			x
[Elkamel et al. 2016]		x	x
[Basciani et al. 2016]		x	
[Bawa and Kaur 2017]		x	
[Kögel 2017]			x
[Le et al. 2018]			x
[Diamantopoulos et al. 2018]	x	x	

Na Tabela 7 é apresentada a relação de quais técnicas foram citadas nos trabalhos e em quais conjuntos de *assets* elas foram aplicadas.

5.2. Mecanismos de recomendação (Q2)

Esta seção responde a **Q2**: Que mecanismos de recomendação são utilizados?

Com relação aos mecanismos de recomendação identificados, não obtivemos nenhum indício de que há uma técnica que se destaque. Ou seja, a área carece de estudos comparativos, principalmente de estudos primários quantitativos. Por exemplo, como pode ser observado na Tabela 8, a única técnica de aprendizado de máquina que foi aplicada em mais de um trabalho foi algoritmo evolutivo. Portanto, visto que tratam-se de estudos exploratórios, trata-se de um indicativo de lacuna de pesquisa, já que existe a necessidade de experimentar diferentes abordagens, bem como realizar estudos comparativos e explicativos.

A maioria dos trabalhos apresentaram alguma forma de validação para suas abordagens, o que abre espaço para algumas observações:

1. **A pesquisa na área carece de estudos experimentais.** A qualidade da abordagem de recomendação pode ser medida através das métricas *Precision* e *Recall*.
   (a) Nos artigos [Heinemann 2012] e [Kögel 2017], os autores usaram *Precision* e *Recall*, com o primeiro estudo mostrando uma precisão de 0.66 e *Recall* de 0.49. O segundo estudo teve como *Precision* 0.43 e *Recall* de 0.82. Sendo *Precision* a relevância dos itens selecionados, e *Recall* a

Tabela 7. Técnicas aplicadas aos tipos de *assets*.

Artigo	Técnica	Tipo de *Assets*
[Ye and Lo 2000]	Self-Organizing Map	Componentes
[Nakkrasae and Sophatsathit 2004]	RPCL(Neural Network)	Componentes
[Li et al. 2004]	Information Entropy Theory	Componentes
[Wang et al. 2004]	Self-Organizing Map	Componentes
[McCarey et al. 2005]	Collaborative filtering, Content-based Filtering	Componentes
[Wu et al. 2007]	Ontologia, text mining	Documentos
[Bajracharya et al. 2009]	Topic Model e Author Topic Model	Source Code
[Martins et al. 2009]	Association Rule	Componentes
[Wang and Ren 2011]	Clustering Index Tree	Componentes
[Dumitru et al. 2011]	Text Mining, K-Nearest-Neighbor, Incremental Diffusive Clustering	Domain-Specific Features
[Heinemann 2012]	Collaborative Filtering e Association Rules	Tool/DSL
[Kumar et al. 2011]	K-Means	Casos de uso em MDL (Rational Rose)
[Sayyad et al. 2012]	Range Ranking Method	Configuração de linha de produtos
[Vodithala and Pabboju 2015]	Clustering	Componente (Funções)
[Vescan 2015]	Algoritmo evolutivo	Componentes
[Elkamel et al. 2016]	CACB	Modelo (Classes UML)
[Basciani et al. 2016]	Hierarchical clustering	Metamodelos
[Bawa and Kaur 2017]	Clustering	Componentes
[Kögel 2017]	Algoritmo evolutivo	Modelos
[Le et al. 2018]	Sequential pattern	Source code
[Diamantopoulos et al. 2018]	K-Means	Source code

Tabela 8. Técnicas usadas para recomendação.

Artigo	Recomendação
[Li et al. 2004]	Information Entropy Theory
[McCarey et al. 2005]	Collaborative Filtering
[Martins et al. 2009]	Association Rule
[Dumitru et al. 2011]	Association Rule e k-Nearest-Neighbor
[Heinemann 2012]	Collaborative Filtering e Association Rule
[Sayyad et al. 2012]	Range Ranking
[Vescan 2015]	Algoritmo Evolutivo
[Elkamel et al. 2016]	CACB
[Kögel 2017]	Algoritmo Evolutivo
[Le et al. 2018]	Sequential Pattern

quantidade de itens relevantes no grupo selecionado, pode-se considerar que [Heinemann 2012] obteve uma melhor seleção de *assets*, enquanto [Kögel 2017] obteve um número maior de *assets* relevantes dentro do grupo de resultado de busca.

(b) O estudo [Martins et al. 2009] apresentou um experimento controlado. O resultado do experimento mostrou uma taxa de sugestão de *assets* de 100%. Mesmo com o resultado positivo, os autores apontaram que mais teste em cenários e dados reais são necessários para corroborar tal resultado.

(c) No artigo de [McCarey et al. 2005], os autores mostraram que a abordagem proposta teve um *Recall* médio de 39.8% e *Precision* média de 22.7%. Os resultados foram considerados promissores comparados a *baseline* prevista.

(d) O número pequeno de contribuições empíricas e quantitativas, é um indicativo de que estudos futuros poderiam explorar as métricas citadas para que seja possível comparar resultados. Sem tais medições, se torna mais

difícil a comparação de abordagens.

2. **Tecnologias antigas podem apresentar bons resultados e lições**. Como exemplo, o estudo [Li et al. 2004], é classificado como recomendação de componente de software e apresentou bons resultados em comparação com repositórios modernos conhecidos que suportam MDE, como MDEForge [Basciani et al. 2016], SEMAT [Jacobson et al. 2012] e SHARE [Gorp and Mazanek 2011]. Os dois últimos introduzidos intencionalmente para fins de discussão e não obtidos através do protocolo executado. O estudo [Li et al. 2004] compara uma abordagem baseada em busca, que é adotado pelos três já mencionados anteriormente, com uma estratégia de seleção de atributo randômico. A estratégia de seleção resultou em um resultado de busca mais eficiente e um conjunto de componentes mais confinado.
3. **Ferramentas de recomendação baseadas em OSLC não foram encontradas.** O OSLC é baseado em *Asset Management Specification* (AMS) e provê meios para mineração de dados e recomendações baseada em *linked data*. Essa limitação na literatura caracteriza oportunidades de pesquisa de recomendação baseada no padrão OSLC.

5.3. Tendências de pesquisa (Q3)

Esta seção responde a **Q3**: Quais são as tendências das pesquisas na área?

Neste sentido, pode-se observar uma distribuição comum do número de estudos por técnicas identificadas, como ilustra a Figura 2. Isto é um indicador de que ainda não foi encontrada/concebida/provada uma abordagem próxima ao considerado ideal para a mineração e recomendação de *assets* de *software*.

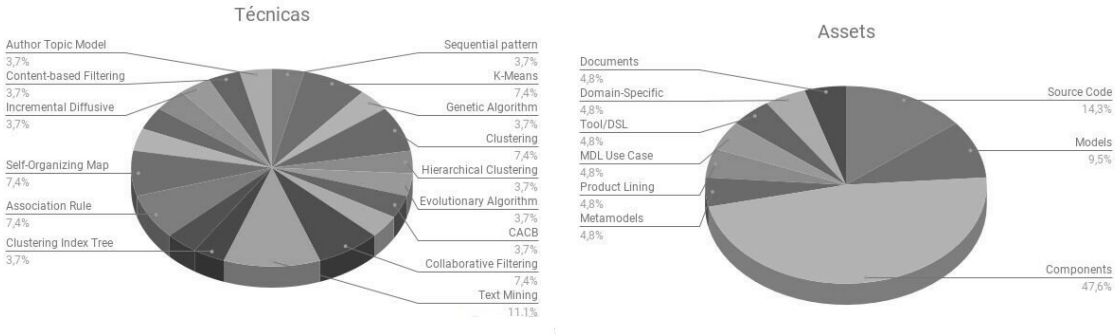

Figura 2. Proporção de técnicas identificadas (a esquerda) e proporção entre os tipos de *assets* identificados (a direita).

Em relação aos tipos de *assets* descritos nos respectivos estudos, encontrou-se que aproximadamente 47% dos artigos abordam *assets* como sendo componentes de software. Demonstrando a diversidade dos *assets*, encontrou-se que aproximadamente 53% dos trabalhos trata de um tipo específico de artefato reutilizável, categorizados em 9 tipos bem definidos. A Figura 2 mostra em detalhes a proporção de *assets* focados nos estudos.

6. Conclusão

O principal objetivo deste trabalho foi relatar os resultados de um mapeamento sistemático de literatura. Com o objetivo de compreender como a área de aprendizado

de máquina da suporte ao reúso oportunista de *assets*, caracterizou-se as técnicas e os processos envolvidos na mineração de *assets* híbridos por meio de questões dirigidas ao suporte de aprendizado de máquina neste contexto. Este trabalho, portanto, apresenta uma análise de como técnicas de recomendação vem contribuindo para a área de reúso oportunista, um estudo singular sobre o estado da arte no tema. Por fim, no âmbito da industria de software e, tendo em vista que ela compartilha do nosso interesse em otimizar os processos de reúso, percebe-se nos estudos analisados duas grandes lacunas para a seleção de opções para recomendação: 1) faltam estudos primários que avaliam as propostas em cenários reais, e 2) a área carece de estudos comparativos de técnicas e também de outros do tipo explicativos que busquem avaliar critérios de adoção na prática.

7. Agradecimentos

Este estudo foi parcialmente financiado através da Pró-Reitoria de Pesquisa (PROPESQ), com bolsa do programa AGP (Apoio a Grupos de Pesquisa), e pela FAPERGS, por meio do projeto ARD N. 19/2551-0001268-3.

Referências

[ams 2012] (2012). Oslc asset management 2.0 specification.

[Bajracharya et al. 2009] Bajracharya, S., Ossher, J., and Lopes, C. (2009). Sourcerer: An internet-scale software repository. In *Proceedings of the 2009 ICSE Workshop on Search-Driven Development-Users, Infrastructure, Tools and Evaluation*, SUITE '09, pages 1–4, Washington, DC, USA. IEEE Computer Society.

[Basciani et al. 2016] Basciani, F., Di Rocco, J., Di Ruscio, D., Iovino, L., and Pierantonio, A. (2016). Automated clustering of metamodel repositories. In *International Conference on Advanced Information Systems Engineering*, pages 342–358. Springer.

[Bawa and Kaur 2017] Bawa, R. K. and Kaur, I. (2017). Classification and clustering for efficient storage and retrieval of component repositories. In *2017 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–6. IEEE.

[Caldiera and Rombach 1994] Caldiera, V. R. B. G. and Rombach, H. D. (1994). The goal question metric approach. *Encyclopedia of software engineering*, pages 528–532.

[Chen et al. 2010] Chen, L., Ali Babar, M., and Zhang, H. (2010). Towards an evidence-based understanding of electronic data sources.

[Diamantopoulos et al. 2018] Diamantopoulos, T., Karagiannopoulos, G., and Symeonidis, A. (2018). Codecatch: extracting source code snippets from online sources. In *2018 IEEE/ACM 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE)*, pages 21–27. IEEE.

[Dumitru et al. 2011] Dumitru, H., Gibiec, M., Hariri, N., Cleland-Huang, J., Mobasher, B., Castro-Herrera, C., and Mirakhorli, M. (2011). On-demand feature recommendations derived from mining public product descriptions. In *Proceedings of the 33rd International Conference on Software Engineering*, ICSE '11, pages 181–190, New York, NY, USA. ACM.

[Elkamel et al. 2016] Elkamel, A., Gzara, M., and Ben-Abdallah, H. (2016). An uml class recommender system for software design. In *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*, pages 1–8. IEEE.

[Fayyad et al. 1996] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37–37.

[Gorp and Mazanek 2011] Gorp, P. V. and Mazanek, S. (2011). Share: a web portal for creating and sharing executable research papers. *Procedia Computer Science*, 4:589 – 597. International Conference on Computational Science, ICCS 2011.

[Heinemann 2012] Heinemann, L. (2012). Facilitating reuse in model-based development with context-dependent model element recommendations. In *Proceedings of the Third International Workshop on Recommendation Systems for Software Engineering*, pages 16–20. IEEE Press.

[Jacobson et al. 2012] Jacobson, I., Ng, P.-W., McMahon, P. E., Spence, I., and Lidman, S. (2012). The essence of software engineering: The semat kernel. a thinking framework in the form of an actionable kernel. *ACMQUEUE. Development 9. Networks*, 10(10):1–12.

[Kögel 2017] Kögel, S. (2017). Recommender system for model driven software development. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2017, pages 1026–1029, New York, NY, USA. ACM.

[Kumar et al. 2011] Kumar, S., Bhatia, R. K., and Kumar, R. (2011). K-means clustering of use-cases using mdl. In *International Conference on Computing and Communication Systems*, pages 57–67. Springer.

[Le et al. 2018] Le, D.-T., Lê, L.-S., Thoai, N., and Nguyen, T.-V. (2018). An old problem with a new therapy: Coupling topic modeling and mining sequential patterns in recommending source code. In *2018 International Conference on Advanced Computing and Applications (ACOMP)*, pages 117–124. IEEE.

[Li et al. 2004] Li, G., Pan, Y., Zhang, L., Xie, B., and Shao, W. (2004). Attribute ranking: An entropy-based approach to accelerating browsing-based component retrieval. In *International Conference on Software Reuse*, pages 232–241. Springer.

[Martins et al. 2009] Martins, A. C., Garcia, V. C., de Almeida, E. S., and d. L. Meira, S. R. (2009). Suggesting software components for reuse in search engines using discovered knowledge techniques. In *2009 35th Euromicro Conference on Software Engineering and Advanced Applications*, pages 412–419.

[McCarey et al. 2005] McCarey, F., Cinnéide, M. Ó., and Kushmerick, N. (2005). Rascal: A recommender agent for agile reuse. *Artificial Intelligence Review*, 24(3):253–276.

[Nakkrasae and Sophatsathit 2004] Nakkrasae, S. and Sophatsathit, P. (2004). An rpcl-based indexing approach for software component classification. *International Journal of Software Engineering and Knowledge Engineering*, 14(05):497–518.

[Petersen et al. 2015] Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1 – 18.

[Pregunta 2007] Pregunta, L. (2007). A estratégia pico para a construção da pergunta de pesquisa e busca de evidências. *Rev Latino-am Enfermagem*, 15(3).

[Sametinger 1997] Sametinger, J. (1997). *Software engineering with reusable components*. Springer Science & Business Media.

[Sayyad et al. 2012] Sayyad, A. S., Ammar, H., and Menzies, T. (2012). Software feature model recommendations using data mining. In *Proceedings of the Third International Workshop on Recommendation Systems for Software Engineering*, RSSE '12, pages 47–51, Piscataway, NJ, USA. IEEE Press.

[Vescan 2015] Vescan, A. (2015). An evolutionary multiobjective approach for the dynamic multilevel component selection problem. In *International Conference on Service-Oriented Computing*, pages 193–204. Springer.

[Vodithala and Pabboju 2015] Vodithala, S. and Pabboju, S. (2015). A clustering technique based on the specifications of software components. In *2015 International Conference on Advanced Computing and Communication Systems*, pages 1–6. IEEE.

[Wang and Ren 2011] Wang, C. and Ren, Y. (2011). A component clustering index tree based on semantic. In *International Conference on Web Information Systems and Mining*, pages 356–362. Springer.

[Wang et al. 2004] Wang, Z., Liu, D., and Feng, X. (2004). Improved som clustering for software component catalogue. In Yin, F.-L., Wang, J., and Guo, C., editors, *Advances in Neural Networks – ISNN 2004*, pages 846–851, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Wu et al. 2007] Wu, Y., Siy, H., Zand, M., and Winter, V. (2007). Construction of ontology-based software repositories by text mining. In Shi, Y., van Albada, G. D., Dongarra, J., and Sloot, P. M. A., editors, *Computational Science – ICCS 2007*, pages 790–797, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Ye and Lo 2000] Ye, H. and Lo, B. W. (2000). A visualised software library: nested self-organising maps for retrieving and browsing reusable software assets. *Neural Computing & Applications*, 9(4):266–279.