Detailed Overview of Software Smells

Tushar Sharma and Diomidis Spinellis

Department of Management Science and Technology
Athens University of Economics and Business
{tushar,dds}@aueb.gr

Abstract

This document provides an overview of literature concerning software smells covering various dimensions of smells along with their corresponding references.

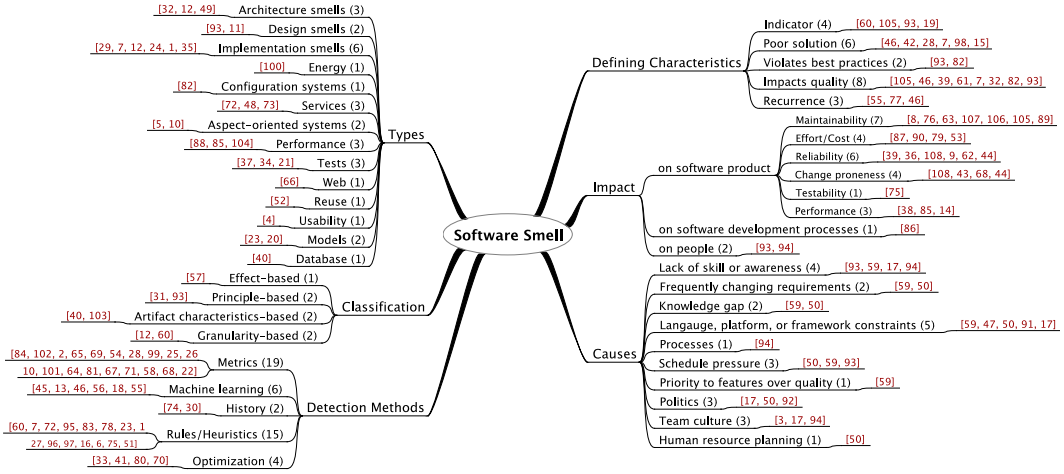

Figure 1: A detailed overview of software smells along with their relevant references.

- Defining characteristics of software smells
  - Indicator [60, 105, 93, 19]
  - Poor solution [46, 42, 28, 7, 98, 15]
  - Violates best practices [93, 82]

  - Impacts quality [105, 46, 39, 61, 7, 32, 82, 93]
  - Recurrence [55, 77, 46]

- Types of software smells
  - Architecture smells [32, 12, 49]
  - Design smells [93, 11]
  - Implementation smells [29, 7, 12, 24, 1, 35]
  - Energy smells [100]
  - Configuration systems smells [82]
  - Services smells [72, 48, 73]
  - Aspect-oriented systems smells [5, 10]
  - Performance smells [88, 85, 104]
  - Test smells [34, 37, 21]
  - Web smells [66]
  - Reuse smells [52]
  - Usability smells [4]
  - Models smells [23, 20]
  - Database smells [40]

- Classification of software smells
  - Effect-based [57]
  - Principle-based [31, 93]
  - Granularity-based [12, 60]
  - Artifact characteristics-based [40, 103]

- Detection methods of software smells
  - Metrics [84, 102, 2, 65, 69, 54, 28, 99, 25, 26, 10, 101, 64, 81, 67, 71, 58, 68, 22]
  - Machine learning [45, 13, 46, 56, 18, 55]
  - History [74, 30]

  - Rules/Heuristics [60, 7, 72, 95, 83, 78, 23, 1, 27, 96, 97, 16, 6, 75, 51]
  - Optimization [33, 41, 80, 70]

- Impact of software smells
  - On software product — maintainability [8, 76, 63, 107, 106, 105, 89], effort/cost [87, 90, 79, 53], reliability [39, 36, 108, 9, 62, 44], change proneness [108, 43, 68, 44], testability [79], performance [38, 85, 14]
  - On software development processes [86]
  - On people [93, 94]

- Causes of software smells
  - Lack of skill or awareness [93, 59, 17, 94]
  - Frequently changing requirements [59, 50]
  - Knowledge gap [59, 50]
  - Language, platform, or framework constraints [59, 47, 50, 91, 17]
  - Processes [94]
  - Schedule pressure [50, 59, 93]
  - Priority to features over quality [59]
  - Politics [17, 50, 92]
  - Team culture [3, 17, 94]
  - Human resource planning [50]

References

[1] Abebe, S. L., Haiduc, S., Tonella, P., Marcus, A., Nov. 2011. The effect of lexicon bad smells on concept location in source code. In: Proceedings - 11th IEEE International Working Conference on Source Code Analysis and Manipulation, SCAM 2011. Fondazione Bruno Kessler, Trento, Italy, IEEE, pp. 125–134.

[2] Abílio, R., Padilha, J., Figueiredo, E., Costa, H., Apr. 2015. Detecting Code Smells in Software Product Lines – An Exploratory Study. In: ITNG '15: Proceedings of the 2015 12th International Conference on Information Technology - New Generations. IEEE Computer Society, pp. 433–438.

[3] Acuña, S. T., Gómez, M., Juristo, N., Aug. 2008. Towards understanding the relationship between team climate and software quality–a quasi-experimental study. Empirical Software Engineering 13 (4), 339–342.

[4] Almeida, D., Campos, J. C., Saraiva, J., Silva, J. C., Apr. 2015. Towards a catalog of usability smells. In: SAC '15: Proceedings of the 30th Annual ACM Symposium on Applied Computing. University of Minho, ACM, pp. 175–181.

[5] Alves, P., Figueiredo, E., Ferrari, F., 2014. Avoiding Code Pitfalls in Aspect-Oriented Programming. In: Computational Science and Its Applications – ICCSA 2012. Springer International Publishing, pp. 31–46.

[6] Arcelli, D., Berardinelli, L., Trubiani, C., Jan. 2015. Performance Antipattern Detection through fUML Model Library. In: WOSP '15: Proceedings of the 2015 Workshop on Challenges in Performance Methods for Software Development. University of L'Aquila, ACM, pp. 23–28.

[7] Arnaoudova, V., Di Penta, M., Antoniol, G., Guéhéneuc, Y.-G., Mar. 2013. A New Family of Software Anti-patterns: Linguistic Anti-patterns. In: CSMR '13: Proceedings of the 2013 17th European Conference on Software Maintenance and Reengineering. IEEE Computer Society, pp. 187–196.

[8] Bavota, G., Qusef, A., Oliveto, R., De Lucia, A., Binkley, D., Dec. 2012. An empirical analysis of the distribution of unit test smells and their impact on software maintenance. In: IEEE International Conference on Software Maintenance, ICSM. Universita di Salerno, Salerno, Italy, IEEE, pp. 56–65.

[9] Bavota, G., Qusef, A., Oliveto, R., De Lucia, A., Binkley, D., May 2014. Are test smells really harmful? An empirical study. Empirical Software Engineering 20 (4), 1052–1094.

[10] Bertran, I. M., Garcia, A., von Staa, A., Mar. 2011. An exploratory study of code smells in evolving aspect-oriented systems. In: AOSD '11: Proceedings of the tenth international conference on Aspect-oriented software development. Pontifical Catholic University of Rio de Janeiro, ACM, p. 203.

[11] Binkley, D., Gold, N., Harman, M., Li, Z., Mahdavi, K., Wegener, J., Dec. 2008. Dependence Anti Patterns. In: Aramis 2008 - 1st International Workshop on Automated engineeRing of Autonomous and runtiMe evolvIng Systems, and ASE2008 the 23rd IEEE/ACM Int. Conf. Automated Software Engineering. King's College London, London, United Kingdom, IEEE, pp. 25–34.

[12] Brown, W. H., Malveau, R. C., McCormick, H. W. S., Mowbray, T. J., 1998. AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis, 1st Edition. John Wiley & Sons, Inc.

[13] Bryton, S., Brito E Abreu, F., Monteiro, M., Dec. 2010. Reducing subjectivity in code smells detection: Experimenting with the Long Method. In: Proceedings - 7th International Conference on the Quality of Information and Communications Technology, QUATIC 2010. Faculdade de Ciencias e Tecnologia, New University of Lisbon, Caparica, Portugal, IEEE, pp. 337–342.

[14] Chen, T.-H., Shang, W., Jiang, Z. M., Hassan, A. E., Nasser, M., Flora, P., May 2014. Detecting performance anti-patterns for applications developed using object-relational mapping. In: ICSE 2014: Proceedings of the 36th International Conference on Software Engineering. Queen's University, Kingston, ACM, pp. 1001–1012.

[15] Cortellessa, V., Di Marco, A., Trubiani, C., Feb. 2014. An approach for modeling and detecting software performance antipatterns based on first-order logics. Software and Systems Modeling (SoSyM) 13 (1), 391–432.

[16] Cortellessa, V., Martens, A., Reussner, R., Trubiani, C., Apr. 2010. A process to effectively identify "guilty" performance antipatterns. In: Lecture Notes in Computer Science (including subseries Lecture Notes

in Artificial Intelligence and Lecture Notes in Bioinformatics). Universita degli Studi dell'Aquila, L'Aquila, Italy, Springer Berlin Heidelberg, pp. 368–382.

[17] Curcio, K., Malucelli, A., Reinehr, S., Paludo, M. A., Nov. 2016. An analysis of the factors determining software product quality: A comparative study. Computer Standards & Interfaces 48, 10–18.

[18] Czibula, G., Marian, Z., Czibula, I. G., Mar. 2015. Detecting software design defects using relational association rule mining. Knowledge and Information Systems 42 (3), 545–577.

[19] da Silva Sousa, L., May 2016. Spotting design problems with smell agglomerations. In: ICSE '16: Proceedings of the 38th International Conference on Software Engineering Companion. Pontifical Catholic University of Rio de Janeiro, ACM, pp. 863–866.

[20] Das, T. K., Dingel, J., Jul. 2016. Model development guidelines for UML-RT: conventions, patterns and antipatterns. Software & Systems Modeling, 1–36.

[21] Deursen, A. V., Moonen, L., Bergh, A. V. D., Kok, G., 2001. Refactoring test code. In: Marchesi, M. (Ed.), Proceedings of the 2nd International Conference on Extreme Programming and Flexible Processes (XP2001). University of Cagliari, pp. 92–95.

[22] Dexun, J., Peijun, M., Xiaohong, S., Tiantian, W., Sep. 2013. Detection and Refactoring of Bad Smell Caused by Large Scale. International Journal of Software Engineering & Applications 4 (5), 1–13.

[23] El-Attar, M., Miller, J., Feb. 2009. Improving the quality of use case models using antipatterns. Software & Systems Modeling 9 (2), 141–160.

[24] Fard, A. M., Mesbah, A., Jan. 2013. JSNOSE: Detecting javascript code smells. In: IEEE 13th International Working Conference on Source Code Analysis and Manipulation, SCAM 2013. The University of British Columbia, Vancouver, Canada, IEEE, pp. 116–125.

[25] Fard, A. M., Mesbah, A., Jan. 2013. JSNOSE: Detecting javascript code smells. In: IEEE 13th International Working Conference on

Source Code Analysis and Manipulation, SCAM 2013. The University of British Columbia, Vancouver, Canada, IEEE, pp. 116–125.

[26] Fenske, W., Schulze, S., Meyer, D., Saake, G., Nov. 2015. When code smells twice as much: Metric-based detection of variability-aware code smells. In: 2015 IEEE 15th International Working Conference on Source Code Analysis and Manipulation, SCAM 2015 - Proceedings. Otto von Guericke University of Magdeburg, Magdeburg, Germany, IEEE, pp. 171–180.

[27] Fokaefs, M., Tsantalis, N., Chatzigeorgiou, A., 2007. JDeodorant: Identification and Removal of Feature Envy Bad Smells. In: 2007 IEEE International Conference on Software Maintenance. Panepistimion Makedonias, Thessaloniki, Greece, IEEE, pp. 519–520.

[28] Fourati, R., Bouassida, N., Abdallah, H. B., 2011. A Metric-Based Approach for Anti-pattern Detection in UML Designs. In: Computer and Information Science 2011. Springer Berlin Heidelberg, pp. 17–33.

[29] Fowler, M., 1999. Refactoring: Improving the Design of Existing Programs, 1st Edition. Addison-Wesley Professional.

[30] Fu, S., Shen, B., Nov. 2015. Code Bad Smell Detection through Evolutionary Data Mining. In: International Symposium on Empirical Software Engineering and Measurement. Shanghai Jiaotong University, Shanghai, China, IEEE, pp. 41–49.

[31] Ganesh, S., Sharma, T., Suryanarayana, G., Jun. 2013. Towards a principle-based classification of structural design smells. Journal of Object Technology 12 (2), 1:1–29.

[32] Garcia, J., Popescu, D., Edwards, G., Medvidovic, N., 2009. Toward a catalogue of architectural bad smells. In: Proceedings of the 5th International Conference on the Quality of Software Architectures: Architectures for Adaptive Software Systems. QoSA '09. Springer-Verlag, pp. 146–162.

[33] Ghannem, A., El Boussaidi, G., Kessentini, M., Mar. 2015. On the use of design defect examples to detect model refactoring opportunities. Software Quality Journal, 1–19.

[34] Greiler, M., van Deursen, A., Storey, M.-A., Jan. 2013. Automated Detection of Test Fixture Strategies and Smells. In: 2013 IEEE Sixth International Conference on Software Testing, Verification and Validation (ICST). IEEE, pp. 322–331.

[35] Guerrouj, L., Kermansaravi, Z., Arnaoudova, V., Fung, B. C. M., Khomh, F., Antoniol, G., Guéhéneuc, Y.-G., May 2016. Investigating the relation between lexical smells and change- and fault-proneness: an empirical study. Software Quality Journal, 1–30.

[36] Hall, T., Zhang, M., Bowes, D., Sun, Y., Sep. 2014. Some Code Smells Have a Significant but Small Effect on Faults. ACM Transactions on Software Engineering and Methodology (TOSEM) 23 (4), 33–39.

[37] Hauptmann, B., Junker, M., Eder, S., Heinemann, L., Vaas, R., Braun, P., May 2013. Hunting for smells in natural language tests. In: ICSE '13: Proceedings of the 2013 International Conference on Software Engineering. Technical University of Munich, IEEE Press, pp. 1217–1220.

[38] Hecht, G., Moha, N., Rouvoy, R., May 2016. An empirical study of the performance impacts of Android code smells. In: MOBILESoft '16: Proceedings of the International Workshop on Mobile Software Engineering and Systems. Universite Lille 2 Droit et Sante, ACM.

[39] Jaafar, F., Guéhéneuc, Y.-G., Hamel, S., Khomh, F., 2013. Mining the relationship between anti-patterns dependencies and fault-proneness. In: Proceedings - Working Conference on Reverse Engineering, WCRE. Ecole Polytechnique de Montreal, Montreal, Canada, IEEE, pp. 351–360.

[40] Karwin, B., 2010. SQL Antipatterns: Avoiding the Pitfalls of Database Programming, 1st Edition. Pragmatic Bookshelf.

[41] Kessentini, W., Kessentini, M., Sahraoui, H., Bechikh, S., Ouni, A., 2014. A Cooperative Parallel Search-Based Software Engineering Approach for Code-Smells Detection. IEEE Transactions on Software Engineering 40 (9), 841–861.

[42] Khan, Y. A., El-Attar, M., 2016. Using model transformation to refactor use case models based on antipatterns. Information Systems Frontiers 18 (1), 171–204.

[43] Khomh, F., Di Penta, M., Guéhéneuc, Y.-G., Dec. 2009. An Exploratory Study of the Impact of Code Smells on Software Change-proneness. In: 2009 16th Working Conference on Reverse Engineering. Ecole Polytechnique de Montreal, Montreal, Canada, IEEE, pp. 75–84.

[44] Khomh, F., Di Penta, M., Guéhéneuc, Y.-G., Antoniol, G., Jun. 2012. An exploratory study of the impact of antipatterns on class change- and fault-proneness. Empirical Software Engineering 17 (3), 243–275.

[45] Khomh, F., Vaucher, S., Guéhéneuc, Y.-G., Sahraoui, H., Aug. 2009. A Bayesian Approach for the Detection of Code and Design Smells. In: QSIC '09: Proceedings of the 2009 Ninth International Conference on Quality Software. IEEE Computer Society, pp. 305–314.

[46] Khomh, F., Vaucher, S., Guéhéneuc, Y.-G., Sahraoui, H., 2011. BDTEX: A GQM-based Bayesian approach for the detection of antipatterns. In: Journal of Systems and Software. Ecole Polytechnique de Montreal, Montreal, Canada, pp. 559–572.

[47] Kleinschmager, S., Hanenberg, S., Robbes, R., Stefik, A., 2012. Do static type systems improve the maintainability of software systems? An empirical study. In: 2012 IEEE 20th International Conference on Program Comprehension (ICPC). Universitat Duisburg-Essen, Essen, Germany, IEEE, pp. 153–162.

[48] Král, J., Žemlička, M., Dec. 2007. The most important service-oriented antipatterns. In: 2nd International Conference on Software Engineering Advances - ICSEA 2007. Charles University in Prague, Prague, Czech Republic, IEEE, pp. 29–29.

[49] Lauder, A., Kent, S., 2000. Legacy System Anti-Patterns and a Pattern-Oriented Migration Response. In: Systems Engineering for Business Process Change. Springer London, pp. 239–250.

[50] Lavallée, M., Robillard, P. N., Aug. 2015. Why good developers write bad code: An observational case study of the impacts of organizational factors on software quality. In: Proceedings - International Conference on Software Engineering. Polytechnique Montréal, Montreal, Canada, IEEE, pp. 677–687.

[51] Ligu, E., Chatzigeorgiou, A., Chaikalis, T., Ygeionomakis, N., Sept 2013. Identification of refused bequest code smells. In: 2013 IEEE International Conference on Software Maintenance. pp. 392–395.

[52] Long, J., Jul. 2001. Software reuse antipatterns. ACM SIGSOFT Software Engineering Notes 26 (4), 68–76.

[53] MacCormack, A., Sturtevant, D. J., 2016. Technical debt and system architecture: The impact of coupling on defect-related activity. Journal of Systems and Software 120, 170 – 182.

[54] Macia, I., Garcia, A., von Staa, A., Dec. 2010. Defining and applying detection strategies for aspect-oriented code smells. In: Proceedings - 24th Brazilian Symposium on Software Engineering, SBES 2010. Pontificia Universidade Catolica do Rio de Janeiro, Rio de Janeiro, Brazil, IEEE, pp. 60–69.

[55] Maiga, A., Ali, N., Bhattacharya, N., Sabané, A., Guéhéneuc, Y.-G., Antoniol, G., Aïmeur, E., Sep. 2012. Support vector machines for anti-pattern detection. In: ASE 2012: Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering. Polytechnic School of Montreal, ACM, pp. 278–281.

[56] Mansoor, U., Kessentini, M., Maxim, B. R., Deb, K., Feb. 2016. Multi-objective code-smells detection using good and bad design examples. Software Quality Journal, 1–24.

[57] Mäntylä, M., Vanhanen, J., Lassenius, C., Sep. 2003. A Taxonomy and an Initial Empirical Study of Bad Smells in Code. In: ICSM '03: Proceedings of the International Conference on Software Maintenance. IEEE Computer Society.

[58] Marinescu, R., Dec. 2005. Measurement and quality in object-oriented design. In: 21st IEEE International Conference on Software Maintenance (ICSM'05). Universitatea Politehnica din Timisoara, Timisoara, Romania, IEEE, pp. 701–704.

[59] Martini, A., Bosch, J., Chaudron, M., Aug 2014. Architecture technical debt: Understanding causes and a qualitative model. In: 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications. pp. 85–92.

[60] Moha, N., Guéhéneuc, Y., Duchien, L., Meur, A. L., 2010. DECOR: A method for the specification and detection of code and design smells. IEEE Trans. Software Eng. 36 (1), 20–36.

[61] Moha, N., Guéhéneuc, Y.-G., 2007. Decor: a tool for the detection of design defects. In: ASE '07: Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering. University of Montreal, ACM, pp. 527–528.

[62] Monden, A., Nakae, D., Kamiya, T., Sato, S.-i., Matsumoto, K.-i., Jun. 2002. Software Quality Analysis by Code Clones in Industrial Legacy Software. In: METRICS '02: Proceedings of the 8th International Symposium on Software Metrics. IEEE Computer Society, p. 87.

[63] Moonen, L., Yamashita, A., Sep. 2012. Do code smells reflect important maintainability aspects? In: ICSM '12: Proceedings of the 2012 IEEE International Conference on Software Maintenance (ICSM). Simula Research Laboratory, IEEE Computer Society.

[64] Munro, M. J., Sep. 2005. Product Metrics for Automatic Identification of "Bad Smell" Design Problems in Java Source-Code. In: METRICS '05: Proceedings of the 11th IEEE International Software Metrics Symposium (METRICS'05). University of Strathclyde, IEEE Computer Society, pp. 15–15.

[65] Murphy-Hill, E., Black, A. P., Oct. 2010. An interactive ambient visualization for code smells. In: SOFTVIS '10: Proceedings of the 5th international symposium on Software visualization. North Carolina State University, ACM.

[66] Nguyen, H. V., Nguyen, H. A., Nguyen, T. T., Nguyen, A. T., Nguyen, T. N., Sep. 2012. Detection of embedded code smells in dynamic web applications. In: ASE 2012: Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering. Iowa State University, ACM, pp. 282–285.

[67] Nongpong, K., Jan. 2015. Feature envy factor: A metric for automatic feature envy detection. In: Proceedings of the 2015-7th International Conference on Knowledge and Smart Technology, KST 2015. Assumption University, Bangkok, Bangkok, Thailand, IEEE, pp. 7–12.

[68] Olbrich, S., Cruzes, D. S., Basili, V., Zazworka, N., Aug. 2009. The evolution and impact of code smells: A case study of two open source systems. In: 2009 3rd International Symposium on Empirical Software Engineering and Measurement (ESEM). IEEE, pp. 390–400.

[69] Oliveto, R., Khomh, F., Antoniol, G., Guéhéneuc, Y.-G., Mar. 2010. Numerical Signatures of Antipatterns: An Approach Based on B-Splines. In: CSMR '10: Proceedings of the 2010 14th European Conference on Software Maintenance and Reengineering. IEEE Computer Society, pp. 248–251.

[70] Ouni, A., Kula, R. G., Kessentini, M., Inoue, K., Jul. 2015. Web Service Antipatterns Detection Using Genetic Programming. In: GECCO '15: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation. Osaka University, ACM, pp. 1351–1358.

[71] Padilha, J., Pereira, J., Figueiredo, E., Almeida, J., Garcia, A., Sant'Anna, C., Jan. 2014. On the effectiveness of concern metrics to detect code smells: An empirical study. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Universidade Federal de Minas Gerais, Belo Horizonte, Brazil, Springer International Publishing, pp. 656–671.

[72] Palma, F., Dubois, J., Moha, N., Guéhéneuc, Y.-G., Jan. 2014. Detection of REST patterns and antipatterns: A heuristics-based approach. In: Franch, X., Ghose, A. K., Lewis, G. A., Bhiri, S. (Eds.), Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Universite du Quebec a Montreal, Montreal, Canada, Springer Berlin Heidelberg, pp. 230–244.

[73] Palma, F., Mohay, N., Jan. 2015. A study on the taxonomy of service antipatterns. In: 2015 IEEE 2nd International Workshop on Patterns Promotion and Anti-Patterns Prevention, PPAP 2015 - Proceedings. Ecole Polytechnique de Montreal, Montreal, Canada, IEEE, pp. 5–8.

[74] Palomba, F., Bavota, G., Di Penta, M., Oliveto, R., Poshyvanyk, D., De Lucia, A., May 2015. Mining version histories for detecting code smells. IEEE Transactions on Software Engineering 41 (5), 462–489.

[75] Palomba, F., Panichella, A., De Lucia, A., Oliveto, R., Zaidman, A., 2016. A textual-based technique for Smell Detection. In: 2016 IEEE 24th International Conference on Program Comprehension (ICPC). Universita di Salerno, Salerno, Italy, IEEE, pp. 1–10.

[76] Perepletchikov, M., Ryan, C., Aug. 2011. A controlled experiment for evaluating the impact of coupling on the maintainability of service-oriented software. IEEE Transactions on Software Engineering 37 (4), 449–465.

[77] Peters, R., Zaidman, A., 2012. Evaluating the lifespan of code smells using software repository mining. In: Proceedings of the 2012 16th European Conference on Software Maintenance and Reengineering. CSMR '12. IEEE Computer Society, pp. 411–416.

[78] Rama, G. M., Feb. 2010. A desiderata for refactoring-based software modularity improvement. In: ISEC '10: Proceedings of the 3rd India software engineering conference. Infosys Technologies Limited India, ACM, pp. 93–102.

[79] Sabané, A., Di Penta, M., Antoniol, G., Guéhéneuc, Y.-G., Mar. 2013. A Study on the Relation between Antipatterns and the Cost of Class Unit Testing. In: CSMR '13: Proceedings of the 2013 17th European Conference on Software Maintenance and Reengineering. IEEE Computer Society, pp. 167–176.

[80] Sahin, D., Kessentini, M., Bechikh, S., Deb, K., Oct. 2014. Code-Smell Detection as a Bilevel Problem. ACM Transactions on Software Engineering and Methodology (TOSEM) 24 (1), 6–44.

[81] Salehie, M., Li, S., Tahvildari, L., Jun. 2006. A Metric-Based Heuristic Framework to Detect Object-Oriented Design Flaws. In: ICPC '06: Proceedings of the 14th IEEE International Conference on Program Comprehension (ICPC'06). University of Waterloo, IEEE Computer Society, pp. 159–168.

[82] Sharma, T., Fragkoulis, M., Spinellis, D., 2016. Does your configuration code smell? In: Proceedings of the 13th International Workshop on Mining Software Repositories. MSR'16. pp. 189–200.

[83] Sharma, T., Mishra, P., Tiwari, R., 2016. Designite — A Software Design Quality Assessment Tool. In: Proceedings of the First International Workshop on Bringing Architecture Design Thinking into Developers' Daily Activities. BRIDGE '16. ACM.

[84] Sharma, V. S., Anwer, S., Dec. 2013. Detecting Performance Antipatterns before Migrating to the Cloud. In: CLOUDCOM '13: Proceedings of the 2013 IEEE International Conference on Cloud Computing Technology and Science - Volume 01. IEEE Computer Society, pp. 148–151.

[85] Sharma, V. S., Anwer, S., Jan. 2014. Performance antipatterns: Detection and evaluation of their effects in the cloud. In: Proceedings - 2014 IEEE International Conference on Services Computing, SCC 2014. Accenture Services Pvt Ltd., India, Bangalore, India, IEEE, pp. 758–765.

[86] Silva, M. C. O., Valente, M. T., Terra, R., 2016. Does technical debt lead to the rejection of pull requests? CoRR abs/1604.01450.

[87] Sjoberg, D. I. K., Yamashita, A., Anda, B., Mockus, A., Dyba, T., Aug. 2013. Quantifying the Effect of Code Smells on Maintenance Effort. IEEE Transactions on Software Engineering 39 (8), 1144–1156.

[88] Smith, C., Dec. 2000. Software performance antipatterns. In: Proceedings Second International Workshop on Software and Performance WOSP 2000. Performance Engineering Services, Santa Fe, United States, pp. 127–136a.

[89] Soh, Z., Yamashita, A., Khomh, F., Guéhéneuc, Y.-G., 2016. Do Code Smells Impact the Effort of Different Maintenance Programming Activities? In: 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER). IEEE, pp. 393–402.

[90] Spínola, R. O., Zazworka, N., Vetrò, A., Seaman, C., Shull, F., 2013. Investigating technical debt folklore: Shedding some light on technical debt opinion. In: Proceedings of the 4th International Workshop on Managing Technical Debt. MTD '13. IEEE Press, pp. 1–7.

[91] Stella, L. F. F., Jarzabek, S., Wadhwa, B., Dec. 2008. A comparative study of maintainability of web applications on J2EE, .NET and ruby

on rails. In: Proceedings - 10th IEEE International Symposium on Web Site Evolution, WSE 2008. National University of Singapore, Singapore City, Singapore, IEEE, pp. 93–99.

[92] Stribrny, S., Mackin, F. B., Sep. 2006. When politics overshadow software quality. IEEE Software 23 (5), 72–73.

[93] Suryanarayana, G., Samarthyam, G., Sharma, T., 2014. Refactoring for Software Design Smells: Managing Technical Debt, 1st Edition. Morgan Kaufmann.

[94] Tom, E., Aurum, A., Vidgen, R., 2013. An exploration of technical debt. Journal of Systems and Software 86 (6), 1498 – 1516.

[95] Trubiani, C., Koziolek, A., Mar. 2011. Detection and solution of software performance antipatterns in palladio architectural models. In: ICPE '11: Proceedings of the 2nd ACM/SPEC International Conference on Performance engineering. Karlsruhe Institute of Technology, ACM, pp. 11–11.

[96] Tsantalis, N., Chaikalis, T., Chatzigeorgiou, A., Apr. 2008. JDeodorant: Identification and Removal of Type-Checking Bad Smells. In: CSMR '08: Proceedings of the 2008 12th European Conference on Software Maintenance and Reengineering. University of Macedonia, IEEE Computer Society, pp. 329–331.

[97] Tsantalis, N., Chatzigeorgiou, A., Oct. 2011. Identification of extract method refactoring opportunities for the decomposition of methods. Journal of Systems & Software 84 (10), 1757–1782.

[98] Van Emden, E., Moonen, L., 2002. Java quality assurance by detecting code smells. Ninth Working Conference on Reverse Engineering, 97–106.

[99] Van Rompaey, B., Du Bois, B., Demeyer, S., Rieger, M., Dec. 2007. On The Detection of Test Smells: A Metrics-Based Approach for General Fixture and Eager Test. IEEE Transactions on Software Engineering 33 (12), 800–817.

[100] Vetr, A., Ardito, L., Procaccianti, G., Morisio, M., 2013. Definition, implementation and validation of energy code smells: an exploratory study on an embedded system. ThinkMind, pp. 34–39.

[101] Vidal, S., Vazquez, H., Díaz-Pace, J. A., Marcos, C., Garcia, A., Oizumi, W., Feb. 2016. JSpIRIT: A flexible tool for the analysis of code smells. In: Proceedings - International Conference of the Chilean Computer Science Society, SCCC. Universidad Nacional del Centro de la Provincia de Buenos Aires, Tandil, Argentina, IEEE, pp. 1–6.

[102] Vidal, S. A., Marcos, C., Díaz-Pace, J. A., 2014. An approach to prioritize code smells for refactoring. Automated Software Engineering 23 (3), 501–532.

[103] Wake, W. C., 2003. Refactoring Workbook, 1st Edition. Addison-Wesley Longman Publishing Co., Inc.

[104] Wang, C., Hirasawa, S., Takizawa, H., Kobayashi, H., May 2014. A Platform-Specific Code Smell Alert System for High Performance Computing Applications. In: IPDPSW '14: Proceedings of the 2014 IEEE International Parallel & Distributed Processing Symposium Workshops. IEEE Computer Society, pp. 652–661.

[105] Yamashita, A., 2014. Assessing the capability of code smells to explain maintenance problems: an empirical study combining quantitative and qualitative data. Empirical Software Engineering 19 (4), 1111–1143.

[106] Yamashita, A., Moonen, L., 2013. Exploring the impact of inter-smell relations on software maintainability: An empirical study. In: Proceedings of the 2013 International Conference on Software Engineering. ICSE '13. IEEE Press, pp. 682–691.

[107] Yamashita, A., Moonen, L., Dec. 2013. To what extent can maintenance problems be predicted by code smell detection? - an empirical study. Information and Software Technology 55 (12), 2223–2242.

[108] Zazworka, N., Shaw, M. A., Shull, F., Seaman, C., May 2011. Investigating the impact of design debt on software quality. In: MTD '11: Proceedings of the 2nd Workshop on Managing Technical Debt. Fraunhofer USA, Inc., ACM, pp. 17–23.