# Appendix 3. Selected primary studies

62. De Jesus, J. S., & De Melo, A. C. V. (2017). Technical debt and the software project characteristics. A repository-based exploratory analysis. *Proceedings - 2017 IEEE 19th Conference on Business Informatics, CBI 2017*, *1*, 444–453. https://doi.org/10.1109/CBI.2017.62

67. Dias, M., Bacchelli, A., Gousios, G., Cassou, D., & Ducasse, S. (2015). Untangling fine-grained code changes. *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering, SANER 2015 - Proceedings*, 341–350. https://doi.org/10.1109/SANER.2015.7081844

68. Digkas, G., Lungu, M., Avgeriou, P., Chatzigeorgiou, A., & Ampatzoglou, A. (2018). How do developers fix issues and pay back technical debt in the Apache ecosystem? *25th IEEE International Conference on Software Analysis, Evolution and Reengineering, SANER 2018 - Proceedings*, *2018*–*March*, 153–163. https://doi.org/10.1109/SANER.2018.8330205

69. Ding, J., Sun, H., Wang, X., & Liu, X. (2018). Entity-level sentiment analysis of issue comments. *Proceedings - International Conference on Software Engineering*, 7–13. https://doi.org/10.1145/3194932.3194935

70. Dit, B., Wagner, M., Wen, S., Wang, W., Linares-Vásquez, M., Poshyvanyk, D., & Kagdi, H. (2014). ImpactMiner: A tool for change impact analysis. *36th International Conference on Software Engineering, ICSE Companion 2014 - Proceedings*, 540–543. https://doi.org/10.1145/2591062.2591064

71. Dittrich, A., Gunes, M. H., & Dascalu, S. (2013). Network analysis of software repositories: Identifying subject matter experts. *Studies in Computational Intelligence*, *424*, 187–198. https://doi.org/10.1007/978-3-642-30287-9\_20

72. Do, L. N. Q., Eichberg, M., & Bodden, E. (2016). Toward an automated benchmark management system. *SOAP 2016 - Proceedings of the 5th ACM SIGPLAN International Workshop on State Of the Art in Program Analysis, Co-Located with PLDI 2016*, 13–17. https://doi.org/10.1145/2931021.2931023

73. Dyer, R., Nguyen, H. A., Rajan, H., & Nguyen, T. N. (2015). Boa: Ultra-large-scale software repository and source-code mining. *ACM Transactions on Software Engineering and Methodology*, *25*(1). https://doi.org/10.1145/2803171

74. Elsen. (2016). Software versioning quality parameters: Automated assessment tools based on the parameters, 0–5. https://doi.org/https://doi.org/10.1109/ICODSE.2016.7936139

75. Elsen, S. (2013). VisGi: Visualizing Git branches. *2013 1st IEEE Working Conference on Software Visualization - Proceedings of VISSOFT 2013*. https://doi.org/10.1109/VISSOFT.2013.6650522

76. Falessi, D., & Reichel, A. (2015). Towards an open-source tool for measuring and visualizing the interest of technical debt. *2015 IEEE 7th International Workshop on Managing Technical Debt, MTD 2015 - Proceedings*, 1–8. https://doi.org/10.1109/MTD.2015.7332618

77. Farias, M., Novais, R., Ortins, P., Colaço, M., & Mendonça, M. (2015). Analyzing distributions of emails and commits from OSS contributors through mining software repositories: An exploratory study. *ICEIS 2015 - 17th International Conference on Enterprise Information Systems, Proceedings*, *2*, 303–310. https://doi.org/10.5220/0005368603030310

78. Feiner, J., & Andrews, K. (2018). RepoVis: Visual Overviews and Full-Text Search in Software Repositories. *Proceedings - 6th IEEE Working Conference on Software Visualization, VISSOFT 2018*, 1–11. https://doi.org/10.1109/VISSOFT.2018.00009

79. Fernandez, A., & Bergel, A. (2018). A Domain-Specific Language to Visualize Software Evolution. *Information and Software Technology*, *98*(May 2017), 118–130. https://doi.org/10.1016/j.infsof.2018.01.005

80. Finlay, J., Pears, R., & Connor, A. M. (2014). Data stream mining for predicting software build outcomes using source code metrics. *Information and Software Technology*, *56*(2), 183–198. https://doi.org/10.1016/j.infsof.2013.09.001

81. Fish, A., Nguyen, T. L., & Song, M. (2017). A code inspection tool by mining recurring changes in evolving software. *SoftwareMining 2017 - Proceedings of the 2017 6th IEEE/ACM International Workshop on Software Mining, Co-Located with ASE 2017*, 48–51. https://doi.org/10.1109/SOFTWAREMINING.2017.8100853

82. Fish, A., Nguyen, T. L., & Song, M. (2018). CloneMap: A Clone-Aware Code Inspection Tool in Evolving Software. *IEEE International Conference on Electro Information Technology*, *2018*–*May*, 368–372. https://doi.org/10.1109/EIT.2018.8500143

83. Flisar, J., & Podgorelec, V. (2018). Enhanced feature selection using word embeddings for self-admitted technical debt identification. *Proceedings - 44th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2018*, 230–233. https://doi.org/10.1109/SEAA.2018.00045

84. Fontana, F. A., Rolla, M., & Zanoni, M. (2014). Capturing software evolution and change through code repository smells. In *International Conference on Agile Software Development* (Vol. 199, pp. 148–165). https://doi.org/10.1007/978-3-319-14358-3\_13

85. Forbes, C., Keivanloo, I., & Rilling, J. (2012). Doppel-code: A clone visualization tool for prioritizing global and local clone impacts. *Proceedings - International Computer Software and Applications Conference*, 366–367. https://doi.org/10.1109/COMPSAC.2012.58

92. Giri, A., Ravikumar, A., Mote, S., & Bharadwaj, R. (2016). Vritthi - A theoretical framework for IT recruitment based on machine learning techniques applied over Twitter, LinkedIn, SPOJ and GitHub profiles. *Proceedings of 2016 International Conference on Data Mining and Advanced Computing, SAPIENCE 2016*, 1–7. https://doi.org/10.1109/SAPIENCE.2016.7684163

93. Gonzalez, D., Santos, J. C. S., Popovich, A., Mirakhorli, M., & Nagappan, M. (2017). A Large-Scale Study on the Usage of Testing Patterns That Address Maintainability Attributes: Patterns for Ease of Modification, Diagnoses, and Comprehension. *IEEE International Working Conference on Mining Software Repositories*, 391–401. https://doi.org/10.1109/MSR.2017.8

94. Goon, A., Wu, Y., Matsushita, M., & Inoue, K. (2017). Evolution of code clone ratios throughout development history of open-source C and C++ programs. *IWSC 2017 - 11th IEEE International Workshop on Software Clones, Co-Located with SANER 2017*. https://doi.org/10.1109/IWSC.2017.7880509

95. Gousios, G., & Spinellis, D. (2014). Conducting quantitative software engineering studies with Alitheia Core. *Empirical Software Engineering*, *19*(4), 885–925. https://doi.org/10.1007/s10664-013-9242-3

96. Greene, G. J., Esterhuizen, M., & Fischer, B. (2017). Visualizing and exploring software version control repositories using interactive tag clouds over formal concept lattices. *Information and Software Technology*, *87*, 223–241. https://doi.org/10.1016/j.infsof.2016.12.001

97. Gupta, M., Sureka, A., & Padmanabhuni, S. (2014). Process mining multiple repositories for software defect resolution from control and organizational perspective. *11th Working Conference on Mining Software Repositories, MSR 2014 - Proceedings*, 122–131. https://doi.org/10.1145/2597073.2597081

98. Gyimesi, P., Gyimesi, G., Tóth, Z., & Ferenc, R. (2015). Characterization of Source Code Defects by Data Mining Conducted on GitHub BT  - Computational Science and Its Applications -- ICCSA 2015. In O. Gervasi, B. Murgante, S. Misra, M. L. Gavrilova, A. M. A. C. Rocha, C. Torre, … B. O. Apduhan (Eds.) (pp. 47–62). Cham: Springer International Publishing.

99. Han, J., & Jung, W. (2014). Extracting communication structure of a development organization from a software repository. *Personal and Ubiquitous Computing*, *18*(6), 1413–1421. https://doi.org/10.1007/s00779-013-0742-3

100. Härtel, J., Heinz, M., & Lämmel, R. (2018). EMF patterns of usage on github. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *10890 LNCS*, 216–234. https://doi.org/10.1007/978-3-319-92997-2\_14

101. Hata, H., Todo, T., Onoue, S., & Matsumoto, K. (2015). Characteristics of sustainable OSS projects: A theoretical and empirical study. *Proceedings - 8th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE 2015*, 15–21. https://doi.org/10.1109/CHASE.2015.9

103. Hebig, R., Quang, T. H., Chaudron, M. R. V., Robles, G., & Fernandez, M. A. (2016). The quest for open source projects that use UML: Mining GitHub. *Proceedings - 19th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2016*, 173–183. https://doi.org/10.1145/2976767.2976778

104. Herzig, K., Just, S., & Zeller, A. (2016). The impact of tangled code changes on defect prediction models. *Empirical Software Engineering*, *21*(2), 303–336. https://doi.org/10.1007/s10664-015-9376-6

105. Hidalgo Suarez, C. G., Bucheli, V. A., Restrepo-Calle, F., & Gonzalez, F. A. (2018). *A strategy based on technological maps for the identification of the state-of-the-art techniques in software development projects: Virtual judge projects as a case study*. *Communications in Computer and Information Science* (Vol. 885). Springer International Publishing. https://doi.org/10.1007/978-3-319-98998-3\_27

106. Hindle, A., Bird, C., Zimmermann, T., & Nagappan, N. (2012). Relating requirements to implementation via topic analysis: Do topics extracted from requirements make sense to managers and developers? In *2012 28th IEEE International Conference on Software Maintenance (ICSM)* (pp. 243–252). https://doi.org/10.1109/ICSM.2012.6405278

107. Hora, A., Silva, D., Valente, M. T., & Robbes, R. (2018). Assessing the threat of untracked changes in software evolution. *Proceedings - International Conference on Software Engineering*, 1102–1113. https://doi.org/10.1145/3180155.3180212

108. Hu, J., Sun, X., Lo, D., & Li, B. (2015). Modeling the evolution of development topics using Dynamic Topic Models. *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering, SANER 2015 - Proceedings*, 3–12. https://doi.org/10.1109/SANER.2015.7081810

109. Huang, Y., Zheng, Q., Chen, X., Xiong, Y., Liu, Z., & Luo, X. (2017). Mining Version Control System for Automatically Generating Commit Comment. *International Symposium on Empirical Software Engineering and Measurement*, *2017*–*Novem*, 414–423. https://doi.org/10.1109/ESEM.2017.56

110. Ieva, C., Gotlieb, A., Kaci, S., & Lazaar, N. (2019). Deploying smart program understanding on a large code base. *Proceedings - 2019 IEEE International Conference on Artificial Intelligence Testing, AITest 2019*, 73–80. https://doi.org/10.1109/AITest.2019.000-4

111. Ihara, A., Kamei, Y., Ohira, M., Hassan, A. E., Ubayashi, N., & Matsumoto, K. I. (2014). Early identification of future committers in open source software projects. *Proceedings - International Conference on Quality Software*, 47–56. https://doi.org/10.1109/QSIC.2014.30

112. Ishio, T., Sakaguchi, Y., Ito, K., & Inoue, K. (2017). Source File Set Search for Clone-And-Own Reuse Analysis. *IEEE International Working Conference on Mining Software Repositories*, 257–268. https://doi.org/10.1109/MSR.2017.19

113. Jaafar, F., Gueheneuc, Y. G., Hamel, S., & Khomh, F. (2013). Mining the relationship between anti-patterns dependencies and fault-proneness. *Proceedings - Working Conference on Reverse Engineering, WCRE*, 351–360. https://doi.org/10.1109/WCRE.2013.6671310

114. Technology, S., & Technology, S. (2015). Distributed software development in an offshore outsourcing project: A case study of source code evolution and quality. https://doi.org/https://doi.org/10.1016/j.infsof.2015.12.005

115. Jarczyk, O., Jaroszewicz, S., Wierzbicki, A., Pawlak, K., & Jankowski-Lorek, M. (2018). Surgical teams on GitHub: Modeling performance of GitHub project development processes. *Information and Software Technology*, *100*(April 2017), 32–46. https://doi.org/10.1016/j.infsof.2018.03.010

116. Jaruchotrattanasakul, T., Yang, X., Makihara, E., Fujiwara, K., & Iida, H. (2016). Open Source Resume (OSR): A Visualization Tool for Presenting OSS Biographies of Developers. *Proceedings - 7th International Workshop on Empirical Software Engineering in Practice, IWESEP 2016*, 57–62. https://doi.org/10.1109/IWESEP.2016.17

117. Javed, O., Villazón, A., & Binder, W. (2019). JUniVerse: Large-scale jUnit-test analysis in the wild. *Proceedings of the ACM Symposium on Applied Computing*, *Part F1477*, 1768–1775. https://doi.org/10.1145/3297280.3297453

118. Jha, A. K., Lee, S., & Lee, W. J. (2019). An empirical study of configuration changes and adoption in Android apps. *Journal of Systems and Software*, *156*, 164–180. https://doi.org/https://doi.org/10.1016/j.jss.2019.06.095

119. Ji, T., Pan, J., Chen, L., & Mao, X. (2018). Identifying Supplementary Bug-fix Commits. *Proceedings - International Computer Software and Applications Conference*, *1*, 184–193. https://doi.org/10.1109/COMPSAC.2018.00031

120. Jiang, J., Lo, D., He, J., Xia, X., Kochhar, P. S., & Zhang, L. (2017). Why and how developers fork what from whom in GitHub. *Empirical Software Engineering*, *22*(1), 547–578. https://doi.org/10.1007/s10664-016-9436-6

121. Jiang, J., Lo, D., Ma, X., Feng, F., & Zhang, L. (2017). Understanding inactive yet available assignees in GitHub. *Information and Software Technology*, *91*, 44–55. https://doi.org/10.1016/j.infsof.2017.06.005

122. Jiang, J., Lo, D., Zheng, J., Xia, X., Yang, Y., & Zhang, L. (2019). Who should make decision on this pull request? Analyzing time-decaying relationships and file similarities for integrator prediction. *Journal of Systems and Software*, *154*, 196–210. https://doi.org/https://doi.org/10.1016/j.jss.2019.04.055

123. Jiang, Q., Peng, X., Wang, H., Xing, Z., & Zhao, W. (2015). Summarizing Evolutionary Trajectory by Grouping and Aggregating relevant code changes. *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering, SANER 2015 - Proceedings*, 361–370. https://doi.org/10.1109/SANER.2015.7081846

124. Jiang, Q., Peng, X., Wang, H., Xing, Z., & Zhao, W. (2017). Understanding systematic and collaborative code changes by mining evolutionary trajectory patterns. *Journal of Software: Evolution and Process*, *29*(3). https://doi.org/10.1002/smr.1840

125. Jiang, R., Chen, Z., Zhang, Z., Pei, Y., Pan, M., & Zhang, T. (2018). Semantics-based code search using input/output examples. *Proceedings - 18th IEEE International Working Conference on Source Code Analysis and Manipulation, SCAM 2018*, 92–102. https://doi.org/10.1109/SCAM.2018.00018

126. Jiang, S., & McMillan, C. (2017). Towards Automatic Generation of Short Summaries of Commits. *IEEE International Conference on Program Comprehension*, 320–323. https://doi.org/10.1109/ICPC.2017.12

127. Joblin, M., Mauerer, W., Apel, S., Siegmund, J., & Riehle, D. (2015). From developer networks to verified communities: A fine-grained approach. *Proceedings - International Conference on Software Engineering*, *1*, 563–573. https://doi.org/10.1109/ICSE.2015.73

128. Joy, A., Thangavelu, S., & Jyotishi, A. (2018). Performance of GitHub Open-Source Software Project: An Empirical Analysis. *Proceedings of 2018 2nd International Conference on Advances in Electronics, Computers and Communications, ICAECC 2018*, 1–6. https://doi.org/10.1109/ICAECC.2018.8479462

129. Just, S., Herzig, K., Czerwonka, J., & Murphy, B. (2016). Switching to Git: The Good, the Bad, and the Ugly. *Proceedings - International Symposium on Software Reliability Engineering, ISSRE*, 400–411. https://doi.org/10.1109/ISSRE.2016.38

130. Kagdi1, H., Gethers2, M., Poshyvanyk2, D., & Hammad3, M. (2014). Assigning change requests to software developers. *Journal of Software: Evolution and Process*, *26*(12), 1172–1192. https://doi.org/https://doi.org/10.1002/smr.530

131. Kakarontzas, G., Constantinou, E., Ampatzoglou, A., & Stamelos, I. (2013). Layer assessment of object-oriented software: A metric facilitating white-box reuse. *Journal of Systems and Software*, *86*(2), 349–366. https://doi.org/10.1016/j.jss.2012.08.041

132. Kalyan, A., Chiam, M., Sun, J., & Manoharan, S. (2016). A Collaborative Code Review Platform for GitHub. *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS*, *0*, 191–196. https://doi.org/10.1109/ICECCS.2016.032

133. Keivanloo, I., & Rilling, J. (2014). Software trustworthiness 2.0 - A semantic web enabled global source code analysis approach. *Journal of Systems and Software*, *89*(1), 33–50. https://doi.org/https://doi.org/10.1016/j.jss.2013.08.030

134. Kikas, R., Dumas, M., & Pfahl, D. (2016). Using dynamic and contextual features to predict issue lifetime in GitHub projects. *Proceedings - 13th Working Conference on Mining Software Repositories, MSR 2016*, 291–302. https://doi.org/10.1145/2901739.2901751

135. Kim, T., Chandra, R., & Zeldovich, N. (2013). Optimizing unit test execution in large software programs using dependency analysis. *Proceedings of the 4th Asia-Pacific Workshop on Systems, APSys 2013*. https://doi.org/10.1145/2500727.2500748

136. Kirbas, S., Caglayan, B., Hall, T., Counsell, S., Bowes, D., Sen, A., & Bener, A. (2017). The relationship between evolutionary coupling and defects in large industrial software. *Journal of Software: Evolution and Process*, *29*(4). https://doi.org/10.1002/smr.1842

137. Kirinuki, H., Higo, Y., Hotta, K., & Kusumoto, S. (2016). Splitting commits via past code changes. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *0*, 129–136. https://doi.org/10.1109/APSEC.2016.028

138. Kirinuki, H., Higo, Y., Hotta, K., & Kusumoto, S. (2014). Hey! Are you committing tangled changes? *22nd International Conference on Program Comprehension, ICPC 2014 - Proceedings*, 262–265. https://doi.org/10.1145/2597008.2597798

139. Kononenko, O., Baysal, O., Guerrouj, L., Cao, Y., & Godfrey, M. W. (2015). Investigating code review quality: Do people and participation matter? *2015 IEEE 31st International Conference on Software Maintenance and Evolution, ICSME 2015 - Proceedings*, 111–120. https://doi.org/10.1109/ICSM.2015.7332457

140. Kovalenko, V., Palomba, F., & Bacchelli, A. (2018). Mining file histories: Should we consider branches? *ASE 2018 - Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, 202–213. https://doi.org/10.1145/3238147.3238169

141. Kowark, T., Matthies, C., Uflacker, M., & Plattner, H. (2016). Lightweight collection and storage of software repository data with datarover. *ASE 2016 - Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 810–815. https://doi.org/10.1145/2970276.2970286

142. Krohn, R., & Weninger, T. (2019). Library adoption in public software repositories. *Journal of Big Data*, *6*(1). https://doi.org/10.1186/s40537-019-0201-8

143. Krüger, J., Mukelabai, M., Gu, W., Shen, H., Hebig, R., & Berger, T. (2019). Where is my feature and what is it about? A case study on recovering feature facets. *Journal of Systems and Software*, *152*, 239–253. https://doi.org/10.1016/j.jss.2019.01.057

146. Kusunoki, N., Hotta, K., Higo, Y., & Kusumoto, S. (2013). How much do code repositories include peripheral modifications? *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *2*, 19–24. https://doi.org/10.1109/APSEC.2013.106

147. Kuutila, M., Mäntylä, M. V., Claes, M., Elovainio, M., & Adams, B. (2018). Using experience sampling to link software repositories with emotions and work well-being. *International Symposium on Empirical Software Engineering and Measurement*. https://doi.org/10.1145/3239235.3239245

148. Lamba, Y., Khattar, M., & Sureka, A. (2015). Pravaaha: Mining android applications for discovering API call usage patterns and trends. *ACM International Conference Proceeding Series*, *18*–*20*–*Febr*, 10–19. https://doi.org/10.1145/2723742.2723743

149. Lavoie, T., Khomh, F., Merlo, E., & Zou, Y. (2012). Inferring repository file structure modifications using nearest-neighbor clone detection. *Proceedings - Working Conference on Reverse Engineering, WCRE*, 325–334. https://doi.org/10.1109/WCRE.2012.42

150. Le, X. B. D. (2016). Towards efficient and effective automatic program repair. *ASE 2016 - Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 876–879. https://doi.org/10.1145/2970276.2975934

151. Lee, A. (2018). One-Time Contributors to FLOSS. *ACM SIGSOFT Software Engineering Notes*, *43*(1), 1–6. https://doi.org/10.1145/3178315.3178327

152. Lee, H. J., Seo, B. K., & Seo, E. (2013). A git source repository analysis tool based on a novel branch-oriented approach. *2013 International Conference on Information Science and Applications, ICISA 2013*, 0–3. https://doi.org/10.1109/ICISA.2013.6579457

153. Lee, J., Jung, W., & Extraction, A. (2013). Analyzing Source Code Repositories, 1–2. https://doi.org/10.1109/ICISA.2013.6579464

154. Leitner, P., & Bezemer, C. P. (2017). An exploratory study of the state of practice of performance testing in Java-based open source projects. *ICPE 2017 - Proceedings of the 2017 ACM/SPEC International Conference on Performance Engineering*, 373–384. https://doi.org/10.1145/3030207.3030213

155. Silveira Lelis, C. A., Fernandes Tavares, J., Pereira Araujo, M. A., & Nazar David, J. M. (2016). GiveMe Trace: A Software Evolution Traceability Support Tool. *IEEE Latin America Transactions*, *14*(7), 3444–3454. https://doi.org/10.1109/TLA.2016.7587653

156. Lesenich, O., Apel, S., Kastner, C., Seibt, G., & Siegmund, J. (2017). Renaming and shifted code in structured merging: Looking ahead for precision and performance. *ASE 2017 - Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering*, 543–553. https://doi.org/10.1109/ASE.2017.8115665

157. Levin, S., & Yehudai, A. (2017). Boosting automatic commit classification into maintenance activities by utilizing source code changes. *ACM International Conference Proceeding Series*, 97–106. https://doi.org/10.1145/3127005.3127016

158. Li, H. Y., Li, M., & Zhou, Z. H. (2019). Towards one reusable model for various software defect mining tasks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11441 LNAI*, 212–224. https://doi.org/10.1007/978-3-030-16142-2\_17

159. Li, J., & Ernst, M. D. (2012). CBCD: Cloned buggy code detector. *Proceedings - International Conference on Software Engineering*, 310–320. https://doi.org/10.1109/ICSE.2012.6227183

160. Li, L., Goethals, F., Baesens, B., & Snoeck, M. (2017). Predicting software revision outcomes on GitHub using structural holes theory. *Computer Networks*, *114*, 114–124. https://doi.org/10.1016/j.comnet.2016.08.024

161. Li, S., Tsukiji, H., & Takano, K. (2016). Analysis of software developer activity on a distributed version control system. *Proceedings - IEEE 30th International Conference on Advanced Information Networking and Applications Workshops, WAINA 2016*, 701–707. https://doi.org/10.1109/WAINA.2016.107

162. Li, Y., Zhu, C., Rubin, J., & Chechik, M. (2018). Semantic Slicing of Software Version Histories. *IEEE Transactions on Software Engineering*, *44*(2), 182–201. https://doi.org/10.1109/TSE.2017.2664824

163. Licorish, S. A., & MacDonell, S. G. (2017). Exploring software developers’ work practices: Task differences, participation, engagement, and speed of task resolution. *Information and Management*, *54*(3), 364–382. https://doi.org/10.1016/j.im.2016.09.005

164. Lima, P., Guerra, E., Meirelles, P., Kanashiro, L., Silva, H., & Silveira, F. F. (2018). A Metrics Suite for code annotation assessment. *Journal of Systems and Software*, *137*, 163–183. https://doi.org/10.1016/j.jss.2017.11.024

165. Linares-Vasquez, M., Hossen, K., Dang, H., Kagdi, H., Gethers, M., & Poshyvanyk, D. (2012). Triaging incoming change requests: Bug or commit history, or code authorship? *IEEE International Conference on Software Maintenance, ICSM*, 451–460. https://doi.org/10.1109/ICSM.2012.6405306

166. Lipčák, J., & Rossi, B. (2018). A large-scale study on source code reviewer recommendation. *Proceedings - 44th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2018*, 378–387. https://doi.org/10.1109/SEAA.2018.00068

167. Liu, J., Li, J., & He, L. (2016). A Comparative Study of the Effects of Pull Request on GitHub Projects. *Proceedings - International Computer Software and Applications Conference*, *1*, 313–322. https://doi.org/10.1109/COMPSAC.2016.27

168. Ludwig, J., Xu, S., & Webber, F. (2017). Compiling static software metrics for reliability and maintainability from GitHub repositories. *2017 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2017*, *2017*–*Janua*, 5–9. https://doi.org/10.1109/SMC.2017.8122569

169. Luo, Z., Mao, X., & Li, A. (2015). An Exploratory Research of GitHub Based on Graph Model. *Proceedings - 2015 9th International Conference on Frontier of Computer Science and Technology, FCST 2015*, 96–103. https://doi.org/10.1109/FCST.2015.45

170. Madeyski, L., & Kawalerowicz, M. (2017). Continuous defect prediction: The idea and a related dataset. *IEEE International Working Conference on Mining Software Repositories*, 515–518. https://doi.org/10.1109/MSR.2017.46

171. Lech Madeyski, and Marek Majchrzak. Software Measurement and Defect Prediction with Depress Extensible Framework. Foundations of Computing and Decision Sciences, vol. 39, no. 4, 2014. doi:10.2478/fcds-2014-0014

172. Maffort, C., Valente, M. T., Terra, R., Bigonha, M., Anquetil, N., & Hora, A. (2016). *Mining architectural violations from version history*. *Empirical Software Engineering* (Vol. 21). https://doi.org/10.1007/s10664-014-9348-2

173. Malheiros, Y., Moraes, A., Trindade, C., & Meira, S. (2012). A source code recommender system to support newcomers. *Proceedings - International Computer Software and Applications Conference*, 19–24. https://doi.org/10.1109/COMPSAC.2012.11

174. Malhotra, R., Bansal, B., Jain, C., & Punia, E. (2018). An automated tool for collection of code attributes for cross project defect prediction. *Proceedings - 2017 2nd International Conference on Man and Machine Interfacing, MAMI 2017*, *2018*–*March*, 1–6. https://doi.org/10.1109/MAMI.2017.8307864

175. Maqsood, J., Eshraghi, I., & Ali, S. S. (2017). Success or failure identification for GitHub’s open source projects. *ACM International Conference Proceeding Series*, 145–150. https://doi.org/10.1145/3034950.3034957

176. Martínez-Torres, M. R., Toral, S. L., Barrero, F. J., & Gregor, D. (2013). A text categorisation tool for open source communities based on semantic analysis. *Behaviour and Information Technology*, *32*(6), 532–544. https://doi.org/10.1080/0144929X.2011.624634

177. Matthies, C., Teusner, R., & Hesse, G. (2019). Beyond Surveys: Analyzing Software Development Artifacts to Assess Teaching Efforts. *Proceedings - Frontiers in Education Conference, FIE*, *2018*–*Octob*, 1–9. https://doi.org/10.1109/FIE.2018.8659205

178. Hindle, A., Bird, C., Zimmermann, T., & Nagappan, N. (2012). Relating requirements to implementation via topic analysis: Do topics extracted from requirements make sense to managers and developers? In *2012 28th IEEE International Conference on Software Maintenance (ICSM)* (pp. 243–252). https://doi.org/10.1109/ICSM.2012.6405278

185. Moura, M. H. D. De, Nascimento, H. A. D. Do, & Rosa, T. C. (2014). Extracting new metrics from version control system for the comparison of software developers. *Proceedings - 28th Brazilian Symposium on Software Engineering, SBES 2014*, 41–50. https://doi.org/10.1109/SBES.2014.25

189. Norikane, T., Ihara, A., & Matsumoto, K. (2018). Do review feedbacks influence to a contributor’s time spent on oss projects? *Proceedings - 2018 IEEE/ACIS 3rd International Conference on Big Data, Cloud Computing, Data Science and Engineering, BCD 2018*, 109–113. https://doi.org/10.1109/BCD2018.2018.00028

203. Rahman, A., & Williams, L. (2019). Source code properties of defective infrastructure as code scripts. *Information and Software Technology*, *112*(October 2018), 148–163. https://doi.org/10.1016/j.infsof.2019.04.013

223. Syed, D., Sessa, J., Henschel, A., & Svetinovic, D. (2016). Data Analysis of Correlation Between Project Popularity and Code Change Frequency Dabeeruddin, *2*, 405–412. https://doi.org/https://doi.org/10.1007/978-3-319-46681-1\_5

225. Thompson, C., & Wagner, D. (2017). A large-scale study of modern code review and security in open source projects. *ACM International Conference Proceeding Series*, 83–92. https://doi.org/10.1145/3127005.3127014

226. Tufano, M., Palomba, F., Bavota, G., Oliveto, R., Penta, M. Di, De Lucia, A., & Poshyvanyk, D. (2017). When and Why Your Code Starts to Smell Bad (and Whether the Smells Go Away). *IEEE Transactions on Software Engineering*, *43*(11), 1063–1088. https://doi.org/10.1109/TSE.2017.2653105

227. Uemura, K., Mori, A., Choi, E., & Iida, H. (2019). Tracking Method-Level Clones and a Case Study. *IWSC 2019 - 2019 IEEE 13th International Workshop on Software Clones*, 27–33. https://doi.org/10.1109/IWSC.2019.8665851

229. Wan, C., Zhu, Z., Zhang, Y., & Chen, Y. (2016). Multi-perspective change impact analysis using linked data of software engineering. *ACM International Conference Proceeding Series*, *18*–*Septemb*, 95–98. https://doi.org/10.1145/2993717.2993729

230. Weicheng, Y., Beijun, S., & Ben, X. (2013). Mining GitHub: Why Commit Stops -- Exploring the Relationship between Developer’s Commit Pattern and File Version Evolution. In *2013 20th Asia-Pacific Software Engineering Conference (APSEC)* (Vol. 2, pp. 165–169). https://doi.org/10.1109/APSEC.2013.133

233. Xiao, L., Yu, Z., Chen, B., & Wang, X. (2017). How Robust Is Your Development Team? *IEEE Software*, *35*(1), 64–71. https://doi.org/10.1109/MS.2017.4541052

235. Yamamori, A., Hagward, A. M., & Kobayashi, T. (2017). Can Developers’ Interaction Data Improve Change Recommendation? *Proceedings - International Computer Software and Applications Conference*, *1*, 128–137. https://doi.org/10.1109/COMPSAC.2017.79

7. Ali, N., Guéhéneuc, Y. G., & Antoniol, G. (2013). Trustrace: Mining software repositories to improve the accuracy of requirement traceability links. *IEEE Transactions on Software Engineering*, *39*(5), 725–741. https://doi.org/10.1109/TSE.2012.71

8. Aljemabi, M. A., & Wang, Z. (2017). Empirical study on the similarity and difference between VCS-DSN and BTS-DSN. *ACM International Conference Proceeding Series*, 30–37. https://doi.org/10.1145/3034950.3034980

9. Allamanis, M., & Sutton, C. (2013). Mining source code repositories at massive scale using language modeling. *IEEE International Working Conference on Mining Software Repositories*, (Iim), 207–216. https://doi.org/10.1109/MSR.2013.6624029

10. Alohaly, M., & Takabi, H. (2017). When Do Changes Induce Software Vulnerabilities? *Proceedings - 2017 IEEE 3rd International Conference on Collaboration and Internet Computing, CIC 2017*, *2017*–*Janua*, 59–66. https://doi.org/10.1109/CIC.2017.00020

11. de Moura Alves, F. V., de Alcântara dos Santos Neto, P., Lira, W. A. L., & de Sousa Ibiapina, I. M. (2018). Analysis of Code Familiarity in Module and Functionality Perspectives. In *Proceedings of the 17th Brazilian Symposium on Software Quality - SBQS* (pp. 41–50). New York, New York, USA: ACM Press. https://doi.org/10.1145/3275245.3275250

12. An, L., Khomh, F., & Guéhéneuc, Y. G. (2018). An empirical study of crash-inducing commits in Mozilla Firefox. *Software Quality Journal*, *26*(2), 553–584. https://doi.org/10.1007/s11219-017-9361-y

13. Arciniegas-Mendez, M., Zagalsky, A., Storey, M. A., & Hadwin, A. F. (2015). Regulation as an enabler for collaborative software development. *Proceedings - 8th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE 2015*, 97–100. https://doi.org/10.1109/CHASE.2015.29

14. Arima, R., Higo, Y., & Kusumoto, S. (2017). Investigation and Detection of Split Commit. *Proceedings - International Computer Software and Applications Conference*, *2*, 268–269. https://doi.org/10.1109/COMPSAC.2017.155

15. Arora, R., & Garg, A. (2018). Analysis of software repositories using process mining. *Smart Innovation, Systems and Technologies*, *78*, 637–643. https://doi.org/10.1007/978-981-10-5547-8\_65

16. Asaduzzaman, M., Ahasanuzzaman, M., Roy, C. K., & Schneider, K. A. (2016). How developers use exception handling in Java? *Proceedings - 13th Working Conference on Mining Software Repositories, MSR 2016*, 516–519. https://doi.org/10.1145/2901739.2903500

17. Asri, I. El, Kerzazi, N., Uddin, G., Khomh, F., & Janati Idrissi, M. A. (2019). An empirical study of sentiments in code reviews. *Information and Software Technology*, *114*(June), 37–54. https://doi.org/10.1016/j.infsof.2019.06.005

18. Avelino, G., Passos, L., Hora, A., & Valente, M. T. (2019). Measuring and analyzing code authorship in 1 + 118 open source projects. *Science of Computer Programming*, *176*, 14–32. https://doi.org/10.1016/j.scico.2019.03.001

19. Badashian, A. S., & Stroulia, E. (2016). Measuring user influence in Github: The million follower fallacy. *Proceedings - 3rd International Workshop on CrowdSourcing in Software Engineering, CSI-SE 2016*, 15–21. https://doi.org/10.1145/2897659.2897663

20. Bala, S., Cabanillas, C., Mendling, J., Rogge-Solti, A., & Polleres, A. (2015). Mining project-oriented business processes. *13th International Conference, BPM 2015*, *9253*, 425–440. https://doi.org/10.1007/978-3-319-23063-4

21. Balogh, G., Antal, G., Beszedes, A., Vidacs, L., Gyimothy, T., & Vegh, A. Z. (2015). Identifying wasted effort in the field via developer interaction data. *2015 IEEE 31st International Conference on Software Maintenance and Evolution, ICSME 2015 - Proceedings*, 391–400. https://doi.org/10.1109/ICSM.2015.7332490

22. Baltes, S., & Diehl, S. (2019). *Usage and attribution of Stack Overflow code snippets in GitHub projects*. *Empirical Software Engineering* (Vol. 24). Empirical Software Engineering. https://doi.org/10.1007/s10664-018-9650-5

23. Bao, L., Lo, D., Xia, X., Wang, X., & Tian, C. (2016). How Android app developers manage power consumption? An empirical study by mining power management commits. *Proceedings - 13th Working Conference on Mining Software Repositories, MSR 2016*, 37–48. https://doi.org/10.1145/2901739.2901748

24. Barr, E. T., Bird, C., Rigby, P. C., Hindle, A., German, D. M., & Devanbu, P. (2012). Cohesive and isolated development with branches. *Proceedings of the 15th International Conference on Fundamental Approaches to Software Engineering*, *7212 LNCS*, 316–331. https://doi.org/10.1007/978-3-642-28872-2\_22

25. Barr, E. T., Brun, Y., Devanbu, P., Harman, M., & Sarro, F. (2014). The plastic surgery hypothesis. *Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, *16*–*21*–*Nove*, 306–317. https://doi.org/10.1145/2635868.2635898

26. Batista, N. A., Brandão, M. A., Alves, G. B., Da Silva, A. P. C., & Moro, M. M. (2017). Collaboration strength metrics and analyses on GitHub. *Proceedings - 2017 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2017*, 170–178. https://doi.org/10.1145/3106426.3106480

27. Bautista, A. M., & Feliu, T. S. (2015). Proceso para extraer defectos de Repositorios de Software. *2015 10th Iberian Conference on Information Systems and Technologies, CISTI 2015*. https://doi.org/10.1109/CISTI.2015.7170552

28. Bavota, G., & Russo, B. (2016). A large-scale empirical study on self-admitted technical debt. *Proceedings - 13th Working Conference on Mining Software Repositories, MSR 2016*, 315–326. https://doi.org/10.1145/2901739.2901742

29. Bayati, S., Parsons, D., Susnjak, T., & Heidary, M. (2015). Big data analytics on large-scale socio-technical software engineering archives. *2015 3rd International Conference on Information and Communication Technology, ICoICT 2015*, 65–69. https://doi.org/10.1109/ICoICT.2015.7231398

30. Baysal, O. (2013). Informing development decisions: From data to information. *Proceedings - International Conference on Software Engineering*, 1407–1410. https://doi.org/10.1109/ICSE.2013.6606729

31. Behnamghader, P., Alfayez, R., Srisopha, K., & Boehm, B. (2017). Towards better understanding of software quality evolution through commit-impact analysis. *Proceedings - 2017 IEEE International Conference on Software Quality, Reliability and Security, QRS 2017*, 251–262. https://doi.org/10.1109/QRS.2017.36

32. Behnamghader, P., Meemeng, P., Fostiropoulos, I., Huang, D., Srisopha, K., & Boehm, B. (2018). A scalable and efficient approach for compiling and analyzing commit history. *International Symposium on Empirical Software Engineering and Measurement*. https://doi.org/10.1145/3239235.3239237

33. Benomar, O., Abdeen, H., Sahraoui, H., Poulin, P., & Saied, M. A. (2015). Detection of Software Evolution Phases Based on Development Activities. *IEEE International Conference on Program Comprehension*, *2015*–*Augus*, 15–24. https://doi.org/10.1109/ICPC.2015.11

34. Bernardi, M. L., Canfora, G., Di Lucca, G. A., Di Penta, M., & Distante, D. (2018). The relation between developers’ communication and fix-Inducing changes: An empirical study. *Journal of Systems and Software*, *140*, 111–125. https://doi.org/10.1016/j.jss.2018.02.065

35. Bettenburg, N., Shang, W., Ibrahim, W. M., Adams, B., Zou, Y., & Hassan, A. E. (2012). An empirical study on inconsistent changes to code clones at the release level. *Science of Computer Programming*, *77*(6), 760–776. https://doi.org/10.1016/j.scico.2010.11.010

36. Biazzini, M., Monperrus, M., & Baudry, B. (2014). On analyzing the topology of commit histories in decentralized version control systems. *Proceedings - 30th International Conference on Software Maintenance and Evolution, ICSME 2014*, 261–270. https://doi.org/10.1109/ICSME.2014.48

37. Bigliardi, L., Lanza, M., Bacchelli, A., Dambros, M., & Mocci, A. (2014). Quantitatively exploring non-code software artifacts. *Proceedings - International Conference on Quality Software*, 286–295. https://doi.org/10.1109/QSIC.2014.31

38. Borges, H., & Tulio Valente, M. (2018). What’s in a GitHub Star? Understanding Repository Starring Practices in a Social Coding Platform. *Journal of Systems and Software*, *146*, 112–129. https://doi.org/10.1016/j.jss.2018.09.016

39. Borges, H., Hora, A., & Valente, M. T. (2016). Predicting the popularity of GitHub repositories. *ACM International Conference Proceeding Series*, (Dcc). https://doi.org/10.1145/2972958.2972966

40. Brandtner, M., Muller, S. C., Leitner, P., & Gall, H. C. (2015). SQA-Profiles: Rule-based activity profiles for Continuous Integration environments. *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering, SANER 2015 - Proceedings*, 301–310. https://doi.org/10.1109/SANER.2015.7081840

41. Breckel, A. (2012). Error mining: Bug detection through comparison with large code databases. *IEEE International Working Conference on Mining Software Repositories*, 175–178. https://doi.org/10.1109/MSR.2012.6224278

42. Cai, H., & Santelices, R. (2015). A comprehensive study of the predictive accuracy of dynamic change-impact analysis. *Journal of Systems and Software*, *103*, 248–265. https://doi.org/10.1016/j.jss.2015.02.018

43. Calefato, F., Lanubile, F., & Vasilescu, B. (2019). A large-scale, in-depth analysis of developers’ personalities in the Apache ecosystem. *Information and Software Technology*, *114*(May), 1–20. https://doi.org/10.1016/j.infsof.2019.05.012

44. Campos, E. C., & Maia, M. D. A. (2017). Common Bug-Fix Patterns: A Large-Scale Observational Study. *International Symposium on Empirical Software Engineering and Measurement*, *2017*–*Novem*, 404–413. https://doi.org/10.1109/ESEM.2017.55

45. Canfora, G., Cerulo, L., Cimitile, M., & Di Penta, M. (2014). *How changes affect software entropy: An empirical study*. *Empirical Software Engineering* (Vol. 19). https://doi.org/10.1007/s10664-012-9214-z

47. Kritikos, A., & Stamelos, I. (2018). *Developer dynamics and syntactic quality of commit messages in OSS projects* (Vol. 525). Springer International Publishing. https://doi.org/10.1007/978-3-319-92375-8

48. Cheluvaraju, B., Nagal, K., & Pasala, A. (2012). Mining software revision history using advanced social network analysis. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *1*(January 2017), 717–720. https://doi.org/10.1109/APSEC.2012.113

49. Chen, H., Huang, Y., Liu, Z., Chen, X., Zhou, F., & Luo, X. (2019). Automatically detecting the scopes of source code comments. *Journal of Systems and Software*, *153*, 45–63. https://doi.org/10.1016/j.jss.2019.03.010

50. Choudhary, G. R., Kumar, S., Kumar, K., Mishra, A., & Catal, C. (2018). Empirical analysis of change metrics for software fault prediction. *Computers and Electrical Engineering*, *67*, 15–24. https://doi.org/10.1016/j.compeleceng.2018.02.043

51. Ciani, A., Minelli, R., Mocci, A., & Lanza, M. (2015). UrbanIt: Visualizing repositories everywhere. *2015 IEEE 31st International Conference on Software Maintenance and Evolution, ICSME 2015 - Proceedings*, 324–326. https://doi.org/10.1109/ICSM.2015.7332479

52. Claes, M., Mäntylä, M. V., Kuutila, M., & Adams, B. (2018). Do programmers work at night or during the weekend? *Proceedings - International Conference on Software Engineering*, 705–715. https://doi.org/10.1145/3180155.3180193

53. Claes, M., Mäntylä, M., Kuutila, M., & Farooq, U. (2018). Towards Identifying Paid Open Source Developers - A Case Study with Mozilla Developers. https://doi.org/10.1145/3196398.3196447

54. Cosentino, V., Cánovas Izquierdo, J. L., & Cabot, J. (2018). GITANA: A software project inspector. *Science of Computer Programming*, *153*, 30–33. https://doi.org/10.1016/j.scico.2017.12.002

55. Cosentino, V., Izquierdo, J. L. C., & Cabot, J. (2015). Assessing the bus factor of Git repositories. *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering, SANER 2015 - Proceedings*, 499–503. https://doi.org/10.1109/SANER.2015.7081864

56. Da Costa, D. A., Kulesza, U., Aranha, E., & Coelho, R. (2014). Unveiling developers contributions behind code commits: An exploratory study. *Proceedings of the ACM Symposium on Applied Computing*, 1152–1157. https://doi.org/10.1145/2554850.2555030

57. Da Maldonado, E. S., Abdalkareem, R., Shihab, E., & Serebrenik, A. (2017). An empirical study on the removal of Self-Admitted Technical Debt. *Proceedings - 2017 IEEE International Conference on Software Maintenance and Evolution, ICSME 2017*, 238–248. https://doi.org/10.1109/ICSME.2017.8

58. Da Silva, R. R. O., Vernier, E. F., Rauber, P. E., Comba, J. L. D., Minghim, R., & Telea, A. C. (2016). Metric evolution maps: Multidimensional attribute-driven exploration of software repositories. *VMV 2016 - Vision, Modeling and Visualization*, 61–68. https://doi.org/10.2312/vmv.20161343

59. Dai, M., Shen, B., Zhang, T., & Zhao, M. (2014). Impact of consecutive changes on later file versions. *ACM International Conference Proceeding Series*, 17–24. https://doi.org/10.1145/2627508.2627512

60. Dam, H. K., Savarimuthu, B. T. R., Avery, D., & Ghose, A. (2015). Mining Software Repositories for Social Norms. *Proceedings - International Conference on Software Engineering*, *2*, 627–630. https://doi.org/10.1109/ICSE.2015.209

61. Datta, S., Majumder, A., & Naidu, K. V. M. (2012). Capacitated team formation problem on social networks. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1005–1013. https://doi.org/10.1145/2339530.2339690

63. De La Torre, G., Robbes, R., & Bergel, A. (2018). Imprecisions diagnostic in source code deltas. *Proceedings - International Conference on Software Engineering*, 492–502. https://doi.org/10.1145/3196398.3196404

64. Destefanis, G., Ortu, M., Bowes, D., Marchesi, M., & Tonelli, R. (2018). On measuring affects of github issues’ commenters. *Proceedings - International Conference on Software Engineering*, (June), 14–19. https://doi.org/10.1145/3194932.3194936

65. Di Bella, E., Sillitti, A., & Succi, G. (2013). A multivariate classification of open source developers. *Information Sciences*, *221*, 72–83. https://doi.org/10.1016/j.ins.2012.09.031

66. Dias, L. F., Steinmacher, I., & Pinto, G. (2018). Who drives company-owned OSS projects: internal or external members? *Journal of the Brazilian Computer Society*, *24*(1). https://doi.org/10.1186/s13173-018-0079-x

86. Foucault, M., Palyart, M., Blanc, X., Murphy, G. C., & Falléri, J. R. (2015). Impact of developer turnover on quality in open-source software. *2015 10th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, ESEC/FSE 2015 - Proceedings*, 829–841. https://doi.org/10.1145/2786805.2786870

224. Tahir, A., Yamashita, A., Licorish, S., Dietrich, J., & Counsell, S. (2018). Can you tell me if it smells?, 68–78. https://doi.org/10.1145/3210459.3210466

228. Verma, D., & Kumar, S. (2017). Prediction of defect density for open source software using repository metrics. *Journal of Web Engineering*, *16*(3–4), 294–311.

236. Yang, D., Martins, P., Saini, V., & Lopes, C. (2017). Stack Overflow in Github: Any Snippets There? *IEEE International Working Conference on Mining Software Repositories*, 280–290. https://doi.org/10.1109/MSR.2017.13

179. McIntosh, S., & Kamei, Y. (2018). Are Fix-Inducing Changes a Moving Target? A Longitudinal Case Study of Just-In-Time Defect Prediction. *IEEE Transactions on Software Engineering*, *44*(5), 412–428. https://doi.org/10.1109/TSE.2017.2693980

180. Menarini, M., Yan, Y., & Griswold, W. G. (2017). Semantics-assisted code review: An efficient tool chain and a user study. *ASE 2017 - Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering*, 554–565. https://doi.org/10.1109/ASE.2017.8115666

181. Menichinelli, M. (2017). A data-driven approach for understanding Open Design. Mapping social interactions in collaborative processes on GitHub. *The Design Journal*, *20*(sup1), S3643–S3658. https://doi.org/10.1080/14606925.2017.1352869

182. Meqdadi, O., Alhindawi, N., Alsakran, J., Saifan, A., & Migdadi, H. (2019). Mining software repositories for adaptive change commits using machine learning techniques. *Information and Software Technology*, *109*(January), 80–91. https://doi.org/10.1016/j.infsof.2019.01.008

190. Novielli, N., Calefato, F., & Lanubile, F. (2018). A gold standard for emotion annotation in stack overflow. *Proceedings - International Conference on Software Engineering*, 14–17. https://doi.org/10.1145/3196398.3196453

191. Novielli, N., Girardi, D., & Lanubile, F. (2018). A benchmark study on sentiment analysis for software engineering research. *Proceedings - International Conference on Software Engineering*, 364–375. https://doi.org/10.1145/3196398.3196403

192. Oliva, G. A., Da Silva, J. T., Gerosa, M. A., Santana, F. W. S., Werner, C. M. L., De Souza, C. R. B., & De Oliveira, K. C. M. (2015). Evolving the system’s Core: A case study on the identification and characterization of key developers in apache ant. *Computing and Informatics*, *34*(3), 678–724.

193. Oliva, G. A., Steinmacher, I., Wiese, I., & Gerosa, M. A. (2013). What can commit metadata tell us about design degradation? *International Workshop on Principles of Software Evolution (IWPSE)*, 18–27. https://doi.org/10.1145/2501543.2501547

194. Oliveira, J., Borges, D., Silva, T., Cacho, N., & Castor, F. (2018). Do android developers neglect error handling? a maintenance-Centric study on the relationship between android abstractions and uncaught exceptions. *Journal of Systems and Software*, *136*(2017), 1–18. https://doi.org/10.1016/j.jss.2017.10.032

195. Oumarou, H., Anquetil, N., Etien, A., Ducasse, S., & Taiwe, K. D. (2015). Identifying the exact fixing actions of static rule violation. *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering, SANER 2015 - Proceedings*, 371–379. https://doi.org/10.1109/SANER.2015.7081847

213. Schall, D. (2014). Who to follow recommendation in large-scale online development communities. *Information and Software Technology*, *56*(12), 1543–1555. https://doi.org/10.1016/j.infsof.2013.12.003

214. Shakiba, A., Green, R., & Dyer, R. (2016). FourD: Do developers discuss design? Revisited. *SWAN 2016 - Proceedings of the 2nd International Workshop on Software Analytics, Co-Located with FSE 2016*, 43–46. https://doi.org/10.1145/2989238.2989244

215. Shahzadeh, A., Khosravi, A., & Nahavandi, S. (2015). *Exploring social contagion in open-source communities by mining software repositories*. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 9492). https://doi.org/10.1007/978-3-319-26561-2\_78

216. Singh, N., & Singh, P. (2018). How Do Code Refactoring Activities Impact Software Developers’ Sentiments? - An Empirical Investigation into GitHub Commits. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *2017*–*Decem*, 648–653. https://doi.org/10.1109/APSEC.2017.79

217. Singh, V. B., Chaturvedi, K. K., Khatri, S., & Sharma, M. (2016). Complexity of the Code Changes and Issues Dependent Approach to Determine the Release Time of Software Product. *Computational Science and Its Applications – ICCSA 2017*, *1603*, 1–8. https://doi.org/10.1007/978-3-319-62404-4

218. Soares, D. M., de Lima Júnior, M. L., Plastino, A., & Murta, L. (2018). What factors influence the reviewer assignment to pull requests? *Information and Software Technology*, *98*(April 2017), 32–43. https://doi.org/10.1016/j.infsof.2018.01.015

219. Sothornprapakorn, S., Hayashi, S., & Saeki, M. (2018). Visualizing a Tangled Change for Supporting Its Decomposition and Commit Construction. *Proceedings - International Computer Software and Applications Conference*, *1*, 74–79. https://doi.org/10.1109/COMPSAC.2018.00018

220. Souza, R., & Silva, B. (2017). Sentiment Analysis of Travis CI Builds. *IEEE International Working Conference on Mining Software Repositories*, 459–462. https://doi.org/10.1109/MSR.2017.27

221. Sowe, S. K., Cerone, A., & Settas, D. (2014). An empirical study of FOSS developers patterns of contribution: Challenges for data linkage and analysis. *Science of Computer Programming*, *91*(PART B), 249–265. https://doi.org/10.1016/j.scico.2013.11.033

222. Sun, X., Yang, H., Xia, X., & Li, B. (2017). Enhancing developer recommendation with supplementary information via mining historical commits. *Journal of Systems and Software*, *134*, 355–368. https://doi.org/10.1016/j.jss.2017.09.021

232. Wu, Y., Kropczynski, J., Shih, P. C., & Carroll, J. M. (2014). Exploring the ecosystem of software developers on github and other platforms. *Proceedings of the ACM Conference on Computer Supported Cooperative Work, CSCW*, 265–268. https://doi.org/10.1145/2556420.2556483

234. Xiong, Y., Meng, Z., Shen, B., & Yin, W. (2017). Mining developer behavior across git hub and stack overflow. *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE*, 578–583. https://doi.org/10.18293/SEKE2017-062

196. Padhye, R., Mani, S., & Sinha, V. S. (2014). A study of external community contribution to Open-source projects on GitHub. *11th Working Conference on Mining Software Repositories, MSR 2014 - Proceedings*, 332–335. https://doi.org/10.1145/2597073.2597113

197. Palomba, F., Panichella, A., Zaidman, A., Oliveto, R., & De Lucia, A. (2018). The Scent of a Smell: An Extensive Comparison between Textual and Structural Smells. *IEEE Transactions on Software Engineering*, *44*(10), 977–1000. https://doi.org/10.1109/TSE.2017.2752171

198. Palomba, F., Zanoni, M., Fontana, F. A., De Lucia, A., & Oliveto, R. (2019). Toward a smell-aware bug prediction model. *IEEE Transactions on Software Engineering*, *45*(2), 194–218. https://doi.org/10.1109/TSE.2017.2770122

199. Papamichail, M., Diamantopoulos, T., & Symeonidis, A. (2016). User-Perceived Source Code Quality Estimation Based on Static Analysis Metrics. *Proceedings - 2016 IEEE International Conference on Software Quality, Reliability and Security, QRS 2016*, 100–107. https://doi.org/10.1109/QRS.2016.22

200. Paruma-Pabón, O. H., González, F. A., Aponte, J., Camargo, J. E., & Restrepo-Calle, F. (2016). Finding relationships between socio-technical aspects and personality traits by mining developer e-mails. *Proceedings - 9th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE 2016*, 8–14. https://doi.org/10.1145/2897586.2897611

201. Pascarella, L., Spadini, D., Palomba, F., Bruntink, M., & Bacchelli, A. (2018). Information needs in contemporary code review. *Proceedings of the ACM on Human-Computer Interaction*, *2*(CSCW). https://doi.org/10.1145/3274404

202. Paula, A. C. De, Guerra, E., Lopes, C. V., Sajnani, H., & Lemos, O. A. L. (2016). An exploratory study of interface redundancy in code repositories. *Proceedings - 2016 IEEE 16th International Working Conference on Source Code Analysis and Manipulation, SCAM 2016*, 107–116. https://doi.org/10.1109/SCAM.2016.31

87. Fu, Y., Yan, M., Zhang, X., Xu, L., Yang, D., & Kymer, J. D. (2015). Automated classification of software change messages by semi-supervised Latent Dirichlet Allocation. *Information and Software Technology*, *57*(1), 369–377. https://doi.org/10.1016/j.infsof.2014.05.017

88. Fung, K. H., Aurum, A., & Tang, D. (2012). Social forking in open source software: An empirical study. *CEUR Workshop Proceedings*, *855*, 50–57.

89. Gamalielsson, J., & Lundell, B. (2014). Sustainability of Open Source software communities beyond a fork: How and why has the LibreOffice project evolved? *Journal of Systems and Software*, *89*(1), 128–145. https://doi.org/10.1016/j.jss.2013.11.1077

90. Geiger, F. X., Malavolta, I., Pascarella, L., Palomba, F., Di Nucci, D., & Bacchelli, A. (2018). A graph-based dataset of commit history of real-world Android apps. *Proceedings - International Conference on Software Engineering*, 30–33. https://doi.org/10.1145/3196398.3196460

91. Ghaleb, T. A., da Costa, D. A., & Zou, Y. (2019). An empirical study of the long duration of continuous integration builds. *Empirical Software Engineering*, *24*(4), 2102–2139. https://doi.org/10.1007/s10664-019-09695-9

1. Abdalkareem, R., Mujahid, S., Shihab, E., & Rilling, J. (2019). Which Commits Can Be CI Skipped? *IEEE Transactions on Software Engineering*, *14*(8), 1–1. https://doi.org/10.1109/tse.2019.2897300

2. Abdalkareem, R., Shihab, E., & Rilling, J. (2017). On code reuse from StackOverflow: An exploratory study on Android apps. *Information and Software Technology*, *88*, 148–158. https://doi.org/10.1016/j.infsof.2017.04.005

3. Abdeen, H., Bali, K., Sahraoui, H., & Dufour, B. (2015). Learning dependency-based change impact predictors using independent change histories. *Information and Software Technology*, *67*, 220–235. https://doi.org/10.1016/j.infsof.2015.07.007

186. Murgia, A., Tourani, P., Adams, B., & Ortu, M. (2014). Do developers feel emotions? An exploratory analysis of emotions in software artifacts. *11th Working Conference on Mining Software Repositories, MSR 2014 - Proceedings*, 262–271. https://doi.org/10.1145/2597073.2597086

187. Negara, S., Codoban, M., Dig, D., & Johnson, R. E. (2014). Mining fine-grained code changes to detect unknown change patterns. *Proceedings - International Conference on Software Engineering*, (1), 803–813. https://doi.org/10.1145/2568225.2568317

188. Nishi, M. A., & Damevski, K. (2018). Scalable code clone detection and search based on adaptive prefix filtering. *Journal of Systems and Software*, *137*, 130–142. https://doi.org/10.1016/j.jss.2017.11.039

4. Aghajani, E., Mocci, A., Bavota, G., & Lanza, M. (2017). The Code Time Machine. *IEEE International Conference on Program Comprehension*, 356–359. https://doi.org/10.1109/ICPC.2017.6

5. Agrawal, K., Amreen, S., & Mockus, A. (2015). Commit quality in five high performance computing projects. *Proceedings - 2015 International Workshop on Software Engineering for High Performance Computing in Science, SE4HPCS 2015*, 24–29. https://doi.org/10.1109/SE4HPCS.2015.11

6. Agrawal, K., Aschauer, M., Thonhofer, T., Bala, S., Rogge-Solti, A., & Tomsich, N. (2016). Resource Classification from Version Control System Logs. *Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOCW*, *2016*–*Septe*, 249–258. https://doi.org/10.1109/EDOCW.2016.7584383

205. Razzaq, S., & Xie, M. (2019). Understanding the Surviving Bugsin Open Source Software through the Community Perspective: Using Bayesian Analysis. *Proceedings - 2019 Amity International Conference on Artificial Intelligence, AICAI 2019*, 494–498. https://doi.org/10.1109/AICAI.2019.8701295

206. Robles, G., Ho-Quang, T., Hebig, R., Chaudron, M. R. V., & Fernandez, M. A. (2017). An extensive dataset of UML models in GitHub. *IEEE International Working Conference on Mining Software Repositories*, 519–522. https://doi.org/10.1109/MSR.2017.48

207. Rosen, C., Grawi, B., & Shihab, E. (2015). Commit guru: Analytics and risk prediction of software commits. *2015 10th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, ESEC/FSE 2015 - Proceedings*, 966–969. https://doi.org/10.1145/2786805.2803183

208. Rusk, D., & Coady, Y. (2014). Location-based analysis of developers and technologies on GitHub. *Proceedings - 2014 IEEE 28th International Conference on Advanced Information Networking and Applications Workshops, IEEE WAINA 2014*, 681–685. https://doi.org/10.1109/WAINA.2014.110

209. Saini, M., & Chahal, K. K. (2018). Change profile analysis of open-source software systems to understand their evolutionary behavior. *Frontiers of Computer Science*, *12*(6), 1105–1124. https://doi.org/10.1007/s11704-016-6301-0

210. Saini, M., Mehmi, S., & Chahal, K. K. (2016). Understanding Open Source Software Evolution Using Fuzzy Data Mining Algorithm for Time Series Data. *Advances in Fuzzy Systems*, *2016*. https://doi.org/10.1155/2016/1479692

102. Hattori, L., D’Ambros, M., Lanza, M., & Lungu, M. (2013). Answering software evolution questions: An empirical evaluation. *Information and Software Technology*, *55*(4), 755–775. https://doi.org/10.1016/j.infsof.2012.09.001

153. Lee, J., & Jung, W. (2013). Automated Metric Visualizations for Analyzing Source Code Repositories. In *2013 International Conference on Information Science and Applications (ICISA)* (pp. 1–2). https://doi.org/10.1109/ICISA.2013.6579464

204. Rastogi, A., Thummalapenta, S., Zimmermann, T., Nagappan, N., & Czerwonka, J. (2017). Ramp-up Journey of New Hires, 107–111. https://doi.org/10.1145/3021460.3021471

211. Salis, V., & Spinellis, D. (2019). RepoFS: File system view of Git repositories. *SoftwareX*, *9*, 288–292. https://doi.org/10.1016/j.softx.2019.03.007

212. Sasaki, M., Matsumoto, S., & Kusumoto, S. (2018). Integrating source code search into git client for effective retrieving of change history. *2018 IEEE 1st International Workshop on Mining and Analyzing Interaction Histories, MAINT 2018 - Proceedings*, *2018*–*Janua*, 12–16. https://doi.org/10.1109/MAINT.2018.8323089

144. Krutz, D. E., Malachowsky, S. A., & Shihab, E. (2015). Examining the effectiveness of using concolic analysis to detect code clones. *Proceedings of the ACM Symposium on Applied Computing*, *13*–*17*–*Apri*, 1610–1615. https://doi.org/10.1145/2695664.2695929

145. Kumar, L., Sripada, S. K., Sureka, A., & Rath, S. K. (2018). Effective fault prediction model developed using Least Square Support Vector Machine (LSSVM). *Journal of Systems and Software*, *137*, 686–712. https://doi.org/https://doi.org/10.1016/j.jss.2017.04.016

183. Michaud, H. M., Guarnera, D. T., Collard, M. L., & Maletic, J. I. (2017). Recovering commit branch of origin from GitHub repositories. *Proceedings - 2016 IEEE International Conference on Software Maintenance and Evolution, ICSME 2016*, 290–300. https://doi.org/10.1109/ICSME.2016.39

184. Miura, K., McIntosh, S., Kamei, Y., Hassan, A. E., & Ubayashi, N. (2016). The Impact of Task Granularity on Co-evolution Analyses. *International Symposium on Empirical Software Engineering and Measurement*, *08*–*09*–*Sept*. https://doi.org/10.1145/2961111.2962607