International Conference on Software Engineering, 2016, pp. 547–558.

[22] B. Zhang, E. Hill, and J. Clause, "Towards automatically generating descriptive names for unit tests," in *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 2016, pp. 625–636.

[23] M. Acharya, T. Xie, J. Pei, and J. Xu, "Mining API Patterns As Partial Orders from Source Code: From Usage Scenarios to Specifications," in *Proceedings of the the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, 2007, pp. 25–34.

[24] R. P. Buse and W. Weimer, "Synthesizing API usage examples," in *Proceedings of the 34th International Conference on Software Engineering*, 2012, pp. 782–792.

[25] K. Sen, D. Marinov, and G. Agha, "CUTE: A concolic unit testing engine for c," in *Proceedings of the 10th European Software Engineering Conference Held Jointly with 13th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2005, p. 263–272.

[26] P. Godefroid, N. Klarlund, and K. Sen, "DART: Directed automated random testing," in *Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2005, p. 213–223.

[27] S. Thummalapenta, T. Xie, N. Tillmann, J. De Halleux, and W. Schulte, "Mseqgen: Object-oriented unit-test generation via mining source code," in *Proceedings of the the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT symposium on The Foundations of Software Engineering*, 2009, pp. 193–202.

[28] C. Pacheco, S. K. Lahiri, M. D. Ernst, and T. Ball, "Feedback-directed random test generation," in *Proceedings of the 29th International Conference on Software Engineering*, 2007, pp. 75–84.

[29] G. Fraser and A. Zeller, "Mutation-driven generation of unit tests and oracles," *IEEE Transactions on Software Engineering*, vol. 38, no. 2, pp. 278–292, 2012.

[30] K. Taneja and T. Xie, "Diffgen: Automated regression unit-test generation," in *Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering*, 2008, pp. 407–410.

[31] C. Csallner and Y. Smaragdakis, "JCrasher: an automatic robustness tester for java," *Software: Practice and Experience*, vol. 34, no. 11, pp. 1025–1050, 2004.

[32] A. Nistor, Q. Luo, M. Pradel, T. R. Gross, and D. Marinov, "Ballerina: Automatic generation and clustering of efficient random unit tests for multithreaded code," in *Proceedings of the 34th International Conference on Software Engineering*, 2012, pp. 727–737.

[33] J. Offutt and A. Abdurazik, "Generating Tests from UML Specifications," in *UML'99 – The Unified Modeling Language*, 1999, pp. 416–429.

[34] A. Marcus and J. I. Maletic, "Recovering documentation-to-source-code traceability links using latent semantic indexing," in *Proceedings of the 25th International Conference on Software Engineering*, 2003, pp. 125–135.

[35] G. Antoniol, G. Canfora, G. Casazza, A. De Lucia, and E. Merlo, "Recovering traceability links between code and documentation," *IEEE Transactions on Software Engineering*, vol. 28, no. 10, pp. 970–983, 2002.

[36] Y. Zhou, R. Gu, T. Chen, Z. Huang, S. Panichella, and H. Gall, "Analyzing APIs Documentation and Code to Detect Directive Defects," in *IEEE/ACM International Conference on Software Engineering*, 2017, pp. 27–37.

[37] E. Ben Charrada, A. Koziolek, and M. Glinz, "Identifying outdated requirements based on source code changes," in *IEEE International Requirements Engineering Conference*, 2012, pp. 61–70.

[38] M. Soeken, R. Wille, and R. Drechsler, "Assisted behavior driven development using natural language processing," in *Proceedings of the International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, 2012, pp. 269–287.

[39] K. Beck, *Test-driven development: by example*. Addison-Wesley, 2003.

[40] JBehave.org. (2017) What is JBehave? [Online]. Available: https://jbehave.org/

[41] M. Allamanis and C. Sutton, "Mining idioms from source code," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2014, pp. 472–483.

[42] Z. Li and Y. Zhou, "PR-Miner: automatically extracting implicit programming rules and detecting violations in large software code," in *Proceedings of the Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, 2005, pp. 306–315.

[43] J. Dong, Y. Zhao, and T. Peng, "A Review of Design Pattern Mining Techniques," *International Journal of Software Engineering and Knowledge Engineering*, vol. 19, no. 06, pp. 823–855, 2009.

[44] A. Pandel, M. Gupta, and A. Tripathi, "DNIT – A new approach for design pattern detection," in *Proceedings of the International Conference on Computer and Communication Technology*, 2010, pp. 545–550.

Mathieu Nassif is a Ph.D. student in Computer Science at McGill University, under the supervision of Martin Robillard. His research focuses on the extract, representation, and manipulation of knowledge in software systems to optimize the contribution of developers to the system. Mathieu received his M.Sc. in Computer Science from McGill University and his B.Sc. in Mathematics from Université de Montréal.

Alexa Hernandez is an incoming M.Sc. student in Computer Science at McGill University. Her research interests include software design, maintenance, and evolution. Alexa received a B.A. in Computer Science at McGill University, where she worked under the supervision of Martin P. Robillard.

Ashvitha Sridharan is a software engineer optimizing the edge network at Shopify. Her research interests include software design, maintenance, and evolution. Sridharan received a B.Sc. Computer Science at McGill University, Montreal, where she worked under the supervision of Martin P. Robillard.

Martin P. Robillard is a Professor of Computer Science at McGill University. His research investigate how to facilitate the discovery and acquisition of technical, design, and domain knowledge to support the development of software systems. He served as the Program Co-Chair for the 20th ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE 2012) and the 39th ACM/IEEE International Conference on Software Engineering (ICSE 2017). He received his Ph.D. and M.Sc. in Computer Science from the University of British Columbia and a B.Eng. from École Polytechnique de Montréal.