# Mario Linares-Vasquez

## List of Publications by Citations



| | | | |
|---|---|---|---|
| 76 papers | 1,870 citations | 26 h-index | 40 g-index |
| 80 ext. papers | 2,602 ext. citations | 2.9 avg, IF | 5.15 L-index |

| # | Paper | IF | Citations |
|---|---|---|---|
| 76 | API change and fault proneness: a threat to the success of Android apps **2013**, | | 139 |
| 75 | Mining energy-greedy API usage patterns in Android apps: an empirical study **2014**, | | 114 |
| 74 | . *IEEE Transactions on Software Engineering*, **2015**, 41, 384-407 | 3.5 | 96 |
| 73 | Toward Deep Learning Software Repositories **2015**, | | 92 |
| 72 | User reviews matter! Tracking crowdsourced reviews to support evolution of successful apps **2015**, | | 84 |
| 71 | How do API changes trigger stack overflow discussions? a study on the Android SDK **2014**, | | 83 |
| 70 | Triaging incoming change requests: Bug or commit history, or code authorship? **2012**, | | 61 |
| 69 | Automatically Discovering, Reporting and Reproducing Android Application Crashes **2016**, | | 60 |
| 68 | On Automatically Generating Commit Messages via Summarization of Source Code Changes **2014**, | | 54 |
| 67 | Optimizing energy consumption of GUIs in Android apps: a multi-objective approach **2015**, | | 50 |
| 66 | Continuous, Evolutionary and Large-Scale: A New Perspective for Automated Mobile App Testing **2017**, | | 47 |
| 65 | On using machine learning to automatically classify software applications into domain categories. *Empirical Software Engineering*, **2014**, 19, 582-618 | 3.3 | 47 |
| 64 | Revisiting Android reuse studies in the context of code obfuscation and library usages **2014**, | | 44 |
| 63 | How developers detect and fix performance bottlenecks in Android apps **2015**, | | 40 |
| 62 | Crowdsourcing user reviews to support the evolution of mobile apps. *Journal of Systems and Software*, **2018**, 137, 143-162 | 3.3 | 39 |
| 61 | Auto-completing bug reports for Android applications **2015**, | | 37 |
| 60 | **2016**, | | 37 |

| | | |
|---|---|---|
| 59 | How do Developers Test Android Applications? **2017**, | 34 |
| 58 | An exploratory analysis of mobile development issues using stack overflow **2013**, | 34 |
| 57 | Categorizing software applications for maintenance **2011**, | 34 |
| 56 | Mining Android App Usages for Generating Actionable GUI-Based Execution Scenarios **2015**, | 33 |
| 55 | ExPort: Detecting and visualizing API usages in large source code repositories **2013**, | 31 |
| 54 | ChangeScribe: A Tool for Automatically Generating Commit Messages **2015**, | 30 |
| 53 | Enabling mutation testing for Android apps **2017**, | 29 |
| 52 | On-demand Developer Documentation **2017**, | 28 |
| 51 | Domain matters: bringing further evidence of the relationships among anti-patterns, application domains, and quality-related metrics in Java mobile apps **2014**, | 28 |
| 50 | CrashScope: A Practical Tool for Automated Testing of Android Applications **2017**, | 26 |
| 49 | Automatically assessing code understandability: How far are we? **2017**, | 25 |
| 48 | Automatically Documenting Unit Test Cases **2016**, | 24 |
| 47 | On automatically detecting similar Android apps **2016**, | 21 |
| 46 | An Empirical Study on Android-Related Vulnerabilities **2017**, | 20 |
| 45 | Software Documentation Issues Unveiled **2019**, | 19 |
| 44 | When and why developers adopt and change software licenses **2015**, | 19 |
| 43 | Generating Reproducible and Replayable Bug Reports from Android Application Crashes **2015**, | 18 |
| 42 | License usage and changes: a large-scale study on gitHub. *Empirical Software Engineering*, **2017**, 22, 1537–1577 | 16 |

| | | | |
|---|---|---|---|
| 41 | A comprehensive model for code readability. *Journal of Software: Evolution and Process*, **2018**, 30, e19581 | | 15 |
| 40 | RCLinker: Automated Linking of Issue Reports and Commits Leveraging Rich Contextual Information **2015**, | | 15 |
| 39 | License Usage and Changes: A Large-Scale Study of Java Projects on GitHub **2015**, | | 15 |
| 38 | Automatically Assessing Code Understandability. *IEEE Transactions on Software Engineering*, **2021**, 47, 595-613 | 3.5 | 13 |
| 37 | How developers micro-optimize Android apps. *Journal of Systems and Software*, **2017**, 130, 1-23 | 3.3 | 12 |
| 36 | MDroid+ **2018**, | | 12 |
| 35 | Data-Driven Solutions to Detect API Compatibility Issues in Android: An Empirical Study **2019**, | | 11 |
| 34 | Enabling Testing of Android Apps **2015**, | | 11 |
| 33 | Supporting evolution and maintenance of Android apps **2014**, | | 11 |
| 32 | ImpactMiner: a tool for change impact analysis **2014**, | | 11 |
| 31 | Urban Transformations and Health: Methods for TrUST-a Natural Experiment Evaluating the Impacts of a Mass Transit Cable Car in Bogot□Colombia. *Frontiers in Public Health*, **2020**, 8, 64 | 6 | 10 |
| 30 | Automated Tagging of Software Projects Using Bytecode and Dependencies (N) **2015**, | | 10 |
| 29 | Documenting database usages and schema constraints in database-centric applications **2016**, | | 10 |
| 28 | Machine Learning-Based Detection of Open Source License Exceptions **2017**, | | 9 |
| 27 | How do Developers Document Database Usages in Source Code? (N) **2015**, | | 9 |
| 26 | Supporting and Accelerating Reproducible Research in Software Maintenance Using TraceLab Component Library **2013**, | | 9 |
| 25 | FUSION **2016**, | | 9 |
| 24 | Multi-Objective Optimization of Energy Consumption of GUIs in Android Apps. *ACM Transactions on Software Engineering and Methodology*, **2018**, 27, 1-47 | 3.3 | 8 |

| | | | |
|---|---|---|---|
| 23 | The Android OS stack and its vulnerabilities: an empirical study. *Empirical Software Engineering*, **2019**, 24, 2056-2101 | 3.3 | 7 |
| 22 | Supporting and accelerating reproducible empirical research in software evolution and maintenance using TraceLab Component Library. *Empirical Software Engineering*, **2015**, 20, 1198-1236 | 3.3 | 7 |
| 21 | GEMMA: Multi-objective Optimization of Energy Consumption of GUIs in Android Apps **2017**, | | 7 |
| 20 | Software documentation **2020**, | | 7 |
| 19 | Investigating types and survivability of performance bugs in mobile apps. *Empirical Software Engineering*, **2020**, 25, 1644-1686 | 3.3 | 5 |
| 18 | A model for measuring agility in small and medium software development enterprises **2012**, | | 5 |
| 17 | Can Everyone use my app? An Empirical Study on Accessibility in Android Apps **2019**, | | 5 |
| 16 | To distribute or not to distribute? **2018**, | | 5 |
| 15 | Unsupervised Software Categorization Using Bytecode **2015**, | | 4 |
| 14 | Mutode: generic JavaScript and Node.js mutation testing tool **2018**, | | 4 |
| 13 | Aiding comprehension of unit test cases and test suites with stereotype-based tagging **2018**, | | 3 |
| 12 | Automated Documentation of Android Apps. *IEEE Transactions on Software Engineering*, **2021**, 47, 204-220 | 3.5 | 3 |
| 11 | Overcoming language dichotomies **2018**, | | 3 |
| 10 | Enabling Mutant Generation for Open- and Closed-Source Android Apps. *IEEE Transactions on Software Engineering*, **2020**, 1-1 | 3.5 | 2 |
| 9 | API compatibility issues in Android: Causes and effectiveness of data-driven detection techniques. *Empirical Software Engineering*, **2020**, 25, 5006-5046 | 3.3 | 2 |
| 8 | MutAPK: Source-Codeless Mutant Generation for Android Apps **2019**, | | 2 |
| 7 | Automated Extraction of Augmented Models for Android Apps **2018**, | | 2 |
| 6 | MutAPK 2.0: a tool for reducing mutation testing effort of Android apps **2020**, | | 1 |

| | | | |
|---|---|---|---|
| 5 | An Empirical Study of i18n Collateral Changes and Bugs in GUIs of Android apps **2020**, | | 1 |
| 4 | Shallow or Deep? An Empirical Study on Detecting Vulnerabilities using Deep Learning **2021**, | | 1 |
| 3 | Automated GUI Testing of Android Apps: From Research to Practice **2016**, | | 1 |
| 2 | Taxonomy of security weaknesses in Java and Kotlin Android apps. *Journal of Systems and Software*, **2022**, 187, 111233 | 3.3 | 0 |
| 1 | Studying eventual connectivity issues in Android apps. *Empirical Software Engineering*, **2022**, 27, 1 | 3.3 | 0 |