Automated Authoring, Onboarding Developers, and Extracting Decision Rationale

Citation for published version (APA):

Carver, J. C., Capilla, R., Serebrenik, A., & Staron, M. (2021). Automated Authoring, Onboarding Developers, and Extracting Decision Rationale. *IEEE Software*, 38(5), 110-112. [9520305].
<https://doi.org/10.1109/MS.2021.3086915>

Document license:
TAVERNE

DOI:
[10.1109/MS.2021.3086915](https://doi.org/10.1109/MS.2021.3086915)

Document status and date:
Published: 08/09/2021

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Automated Authoring, Onboarding Developers, and Extracting Decision Rationale

Jeffrey C. Carver, Rafael Capilla, Alexander Serebrenik, and Miroslaw Staron

THE “PRACTITIONERS’ DIGEST” department in this issue of *IEEE Software* includes papers from the 2021 International Conference on Software Engineering (ICSE). Feedback or suggestions are welcome. In addition, if you try or adopt any of the practices included in this article, please send me and the authors of the paper(s) a note about your experiences.

Automated Authoring of Methods

“Siri, Write the Next Method,” by Fengcai Wen and colleagues, expands upon the long-standing concept of code completion to include the automation of function signatures and function bodies. Code completion, an important support mechanism used by professional developers, helps complete method names, suggests the signature of an application

programming interface function, or simply formats code. FeaRS, the authors’ new tool, takes that concept one step further by writing entire function signatures and function bodies based on the analysis of 2.5 million commits of existing functions in various open source projects. The evaluation of FeaRS showed its ability in providing highly relevant recommendations for developers of Android apps but also highlighted the limitations of the tool in supporting a wide diversity of implementation scenarios. The best results are obtained for small techniques, with some promising results for GUI-related methods. The authors also have created a plug-in for Android Studio, which should help improve the productivity of professional software engineers. This paper appears in the main research track at ICSE 2021. Access it at <http://bit.ly/PD-2021> -Sept-01.

Onboarding Developers

“A Case Study of Onboarding in Software Teams: Tasks and Strategies,” by An Ju and colleagues, reports on the results of a study with 32 developers and 15 managers involved in the onboarding process. Onboarding new team members requires patience and energy. The new team members need to understand the product, the organization, and the culture of the company. The current managers and team members must understand what the new team members need to be able to function well and to provide value to the product or service. This study identified three strategies that work well in this context: 1) Simple-Complex, when the new developers receive tasks that gradually increase in complexity; 2) Priority-First, when the tasks are assigned by matching priority with seniority; and 3) Exploration-Based, when the uncertain or undefined tasks are assigned to

Digital Object Identifier 10.1109/MS.2021.3086915
Date of current version: 20 August 2021

senior developers to enable exploration of the product structure. The paper breaks down these strategies into tangible guidelines that can be used by other teams. For example, in the Simple-Complex strategy, the paper provides guidelines about three levels of complexity—bugs or configuration setting during the first week, simple features for the next two to three weeks, and more advanced bugs and features during the next three to nine months—eventually leading to expertise in the product. In addition to the case study results, the paper reports on a follow-up survey of 189 developers and 37 managers that found the results to be representative and the recommendations actionable. This paper appears in the main research track at ICSE 2021. Access the paper at <http://bit.ly/PD-2021-Sept-02>.

“Onboarding Versus Diversity, Productivity and Quality—Empirical Study of the OpenStack Ecosystem,” by Armstrong Foundjem and colleagues, describes a study of onboarding in OpenStack, a large-scale, open source software initiative comprising many independent but interrelated projects. To be successful, onboarding in such initiatives, known as *ecosystems*, should support the contributors in obtaining knowledge and mastering skills required by both the ecosystem as a whole as well as individual projects in which the contributor is interested. Based on an observational study of 72 contributors and 13 mentors, the authors have concluded that the mentors balance theoretical (for example, a knowledge of the contribution workflow or the importance of active communication skills; 40%) and hands-on (60%) activities. They further identified primary challenges associated with onboarding: while contributors

should be self-motivated and committed, the mentors should have a vast knowledge of the ecosystem and be ready to invest substantial effort in preparing the teaching material and keeping it up to date. The expected benefits of the onboarding program for the ecosystem are more diverse, productive, and confident contributors as well as a proliferation of best practices and growth. To evaluate whether these expectations correspond to reality, the authors

other ecosystems and companies to invest in onboarding. This paper appears in the main research track of ICSE 2021. Access it at <http://bit.ly/PD-2021-Sept-03>.

Extraction Rationale for Development Decisions

“Extracting Rationale for Open Source Software Development Decisions—A Study of Python Email Archives,” by Pankajeshwara Sharma and colleagues, investigates the hidden

In an evaluation of the new approach, the authors found that 42% of the participants in Clickworker who stated they had programming skills were not able to adequately answer the screening questions.

quantitatively analyzed 84 months of historical code changes, issue reports, and code reviews. Through this study, the authors confirmed that 65% more women and nonbinary developers have participated in the onboarding program as opposed to those who chose not to participate. Moreover, developers who participated in the onboarding program are more versatile in their technical skills and are more likely to be hired. Onboarding increases the contributors' productivity and retention rate, reducing both the effort required from the contributors to get their work accepted and the number of bugs in their contributions. As such, the onboarding program at OpenStack can be seen as highly successful and might encourage

rationale behind the decisions that lead to the maintenance and evolution of software products and how to extract this rationale to promote transparency and accountability. The paper presents Rationale Miner, a tool that uses heuristics for rationale extraction in open source software projects. To extract the rationale, the authors analyze potential decisions captured in the Python email archives stored in GitHub and from 248 Python Enhancement Proposals (PEPs) that include their states (accepted and rejected). Rationale Miner uses several heuristics to analyze thousands of emails and extract the ground truth and patterns for statements of rationale. From the total data set containing around 1.5 million email messages, the authors manually

Onboarding increases the contributors' productivity and retention rate, reducing both the effort required from the contributors to get their work accepted and the number of bugs in their contribution.

extracted 300 rationale sentences related to 259 PEP messages. In addition, the Rationale Miner tool uses 13 heuristics to extract this rationale automatically from PEP messages. The approach facilitates the extraction of rationale stored by Python developers and decision makers, which can be useful for other open source software communities. This paper appears in the main research track at ICSE 2021. Access it at <http://bit.ly/PD-2021-Sept-04>.

Identifying Candidate Developers

"Do You Really Code? Designing and Evaluating Screening Questions for Online Surveys with Programmers," by Anastasia Danilova and colleagues, addresses a problem faced by researchers when recruiting potential participants for surveys or research studies. Based on their analysis, the authors find that many candidates for participation in research studies or surveys are not qualified based on their actual programming skill. Platforms like Clickworker and Qualtrics do not provide mechanisms to help researchers filter out ineligible participants. To help with this problem, the paper describes a set of screening questions that can easily be answered by people who have sufficient programming skill. The authors were guided by four principles when developing these questions: effectiveness, efficiency, robustness against cheating, and language independence. In an evaluation of the new approach, the authors found that 42% of the participants in Clickworker who stated they had programming skills were not able to adequately answer the screening questions. This paper appears in the main research track at ICSE 2021. Access it at <http://bit.ly/PD-2021-Sept-05>.

ABOUT THE AUTHORS

JEFFREY C. CARVER is a professor in the University of Alabama's Department of Computer Science, Tuscaloosa, Alabama, 35487, USA. Further information about him can be found at <http://carver.cs.ua.edu>. Contact him at carver@cs.ua.edu.

RAFAEL CAPILLA is a professor at Rey Juan Carlos University, Madrid, 28933, Spain. Further information about him can be found at <https://www.linkedin.com/in/rafael-capilla-89b3857/>. Contact him at rafael.capilla@urjc.es.

ALEXANDER SEREBRENIK is a professor at Eindhoven University of Technology, Eindhoven, 5600MB, The Netherlands. Further information about him can be found at <https://www.win.tue.nl/~aserebre/>. Contact him at a.serebrenik@tue.nl.

MIROSLAW STARON is a professor in the Software Engineering Division, Chalmers University of Technology and the University of Gothenburg, Gothenburg, SE-412 96, Sweden. Further information about him can be found at website: <https://metricsnas.myqnapcloud.com:8081/WordPress/>. Contact him at mirosław.staron@cse.gu.se.