A Systematic Categorization and Comparison of Approaches and Tools for the Quality Assurance of Jupyter Notebooks

Bachelor Thesis

# Organization

|  |  |
| --- | --- |
| Examiner: | Prof. Dr. Stefan Wagner |
| Supervisors: | Dr. Justus Bogner |
| Student: |  |
| Timeframe: | 2021-10-11 – 2022-04-11 |

# Context & Motivation

Jupyter Notebooks are a popular tool for scientific work among data scientists [1]. With such notebooks, the work may be easier to understand [2]. Authors have the ability to combine images, Markdown-based text, and code. Many use them as an exploratory tool for data science or machine learning [3]. However, authors do not strictly adhere to the rules for clean code when writing these notebooks [3][4]. Even though the code may be not easy to read and may contain errors, some of the code goes directly into production [3]. In a production application, low code quality may have a negative influence on quality attributes like functional correctness or maintainability. Furthermore, a great potential advantage of such notebooks is that they are easily reproducible if written correctly. However, reproducibility is often neglected [4]. Lastly, not only the code itself is valuable, but so are the lessons learned using Jupyter Notebooks. It is not uncommon to use Jupyter Notebooks for teaching [3]. Bad code in such notebooks is even more dangerous since it conveys bad practices to students. There is already some work with proposals to improve the quality and reproducibility of these notebooks, as shown in [5]. Some known tools for quality assurance are: ReproduceMeGit [6], nbQA [7][8], nbval [9]. However, it is difficult to get an overview of the available approaches and tools for the quality assurance of Jupyter Notebooks.

# Objectives

The goal of this study is to systematically compare and categorize tools and approaches for the quality assurance of Jupyter notebooks. The categorization may consider various aspects, such as the targeted quality attributes, programming languages, or tool applicability. The comparison should highlight potential strengths and weaknesses of the approaches and tools, and support researchers and practitioners in choosing suitable quality assurance techniques for their own work. Additionally, an analysis of the practical applicability and usability of selected tools and approaches should be conducted.

# Methods

This thesis aims to identify and analyze the majority of research papers and tools on the quality assurance of Jupyter Notebooks. Therefore, the most suitable method is a systematic literature review [10]. The comparison will be based on a well-defined methodology and search strategy [10]. Practical applicability and usability will be reviewed by applying selected quality assurance tools to some popular Jupyter Notebooks from GitHub [11].

# References

[1] J. M. Perkel, “Why jupyter is data scientists’ computational notebook of choice,” Nature, vol. 563, no. 7729, pp. 145–146, oct 2018.

[2] SHEN, Helen. Interactive notebooks: Sharing the code. Nature News, 2014, 515. Jg., Nr. 7525, S. 151.

[3] KERY, Mary Beth, et al. The story in the notebook: Exploratory data science using a literate programming tool. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 2018. S. 1-11.

[4] WANG, Jiawei; LI, Li; ZELLER, Andreas. Better code, better sharing: on the need of analyzing jupyter notebooks. In: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results. 2020. S. 53-56.

[5] RULE, Adam, et al. Ten simple rules for reproducible research in Jupyter notebooks. arXiv preprint arXiv:1810.08055, 2018.

[6] SAMUEL, Sheeba; KÖNIG-RIES, Birgitta. ReproduceMeGit: a visualization tool for analyzing reproducibility of Jupyter Notebooks. arXiv preprint arXiv:2006.12110, 2020.

[7] <https://github.com/nbQA-dev/nbQA>

[8] <https://nbqa.readthedocs.io/en/latest/index.html>

[9] <https://github.com/computationalmodelling/nbval>

[10] B. Kitchenham and S. Charters, “Guidelines for performing Systematic Literature reviews in Software Engineering,” Keele, UK, 2007.

[11] <https://github.com/>