Kommunikation von Refactorings in Python-Open-Source- Projekten

Frieso Ritter

frieso-ritter@gmx.de

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG
FAKULTÄT
FÜR MATHEMATIK, INFORMATIK UND NATURWISSENSCHAFTEN

Einleitung

- Refactorings sind in der Softwareentwicklung gängige Praxis [4]. Sie werden häufig über entsprechende Einträge in Commit-Nachrichten kommuniziert. Wir untersuchten, auf welche Weise und anhand welcher Muster in der Python-Community über Refactorings kommuniziert wird [5].
- **Refactoringcommit** ist die Bezeichnung für einen Commit, bei dem Refactoringtätigkeiten im Code vorgenommen wurden.

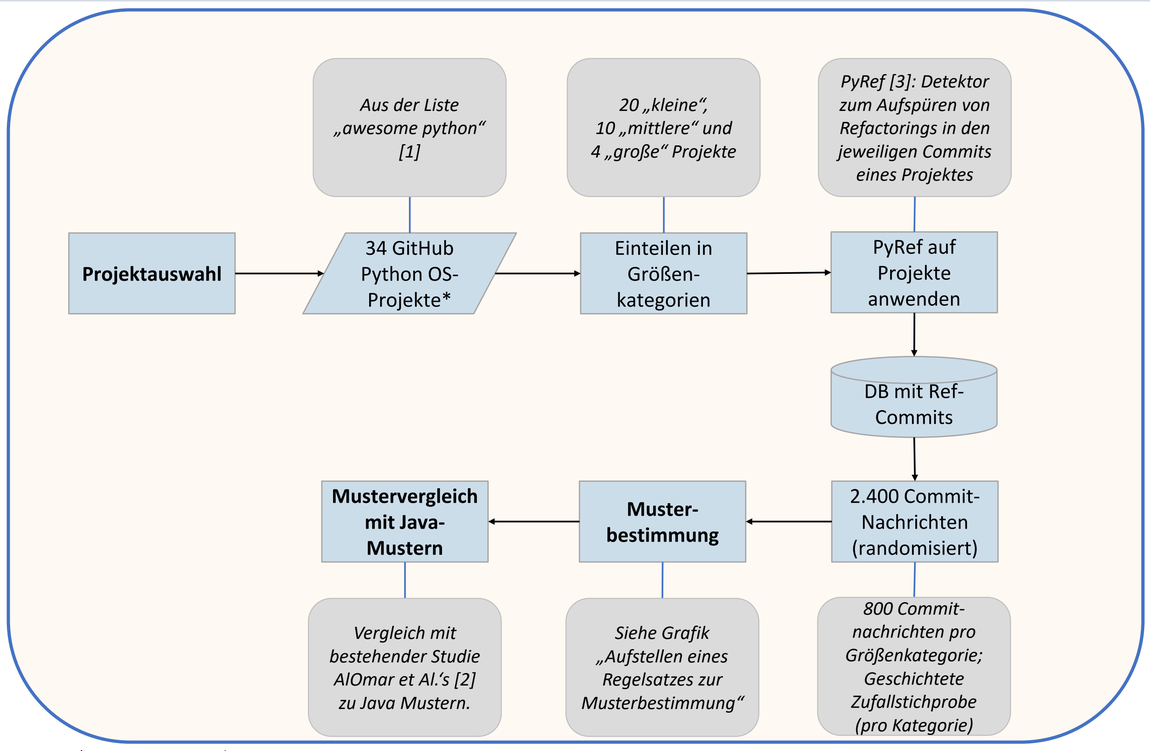
Untersuchungsgegenstand

- **Teil A:** Kommunizieren Python-Entwickler über getätigte Refactoringaktivitäten und falls ja, welche Begriffe verwenden sie bei der Kommunikation von Refactoringaktivitäten?
- **Teil B:** Welche Gemeinsamkeiten und Unterschiede bestehen zwischen den Kommunikationsmustern von Refactorings in Java und Python?

Untersuchungsablauf

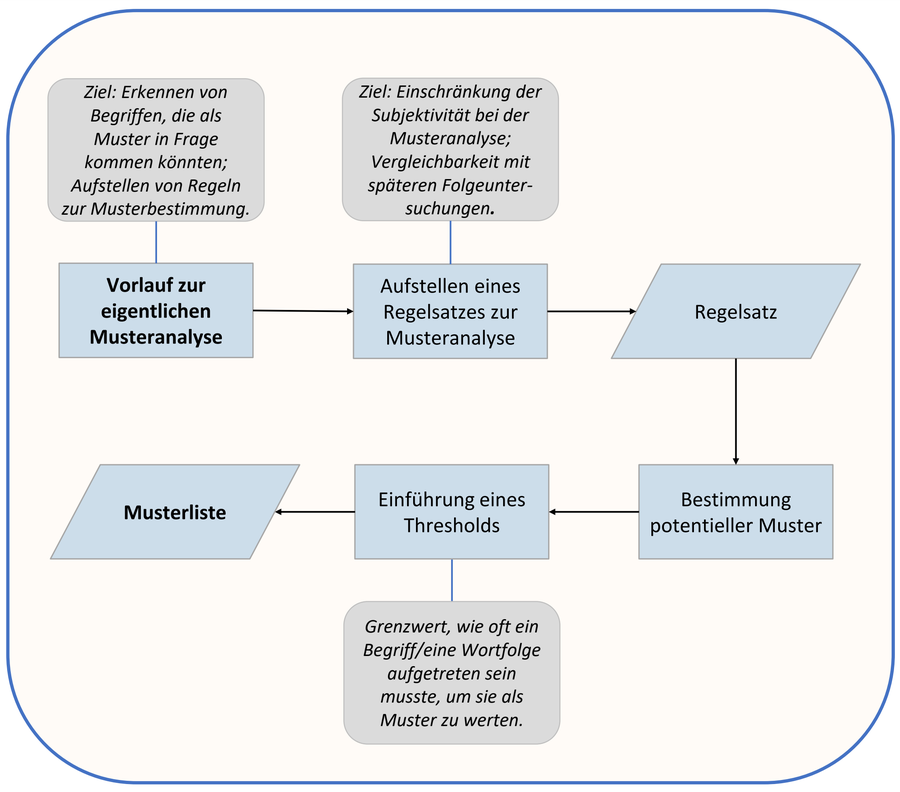
- Wir untersuchten 2400 Commit-Nachrichten, für die Refactoringaktivität im zugehörigen Commit festgestellt werden konnte, auf auftretende Kommunikationsmuster, welche in Zusammenhang mit Refactoringaktivität stehen.

Schematische Darstellung des Untersuchungsablaufs



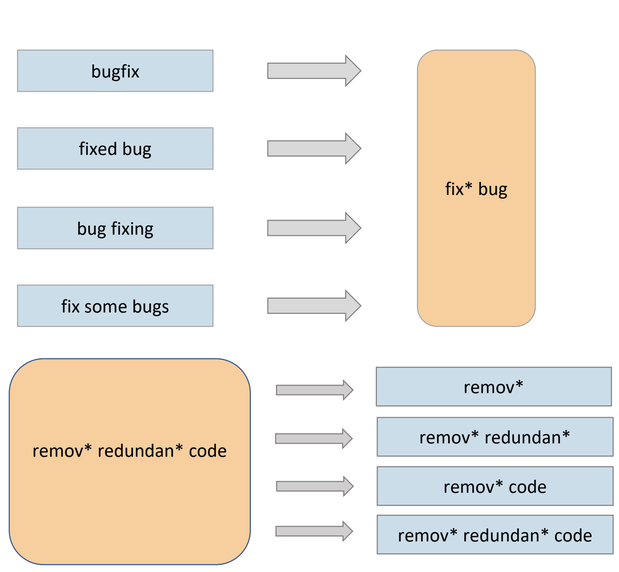
* Open-Source-Projekte

Aufstellen eines Regelsatzes zur Musterbestimmung



Wichtigste Regeln zur Musterbestimmung

- Berücksichtigung der Intention einer Änderung; Beschreibungen, die lediglich die Wirkungsweise eines Elements betreffen, blieben unberücksichtigt.
- Ein Muster wurde pro Commit-Nachricht nur einmal geführt, auch wenn es in der Commit-Nachricht mehrfach vorkam.
- Eng verwandte Wortkombinationen mit gleicher Intention wurden zu einem Muster zusammengeführt.
- Enthielt ein Muster eines oder mehrere Submuster, wurden meist auch die enthaltenen Submuster mitberücksichtigt.

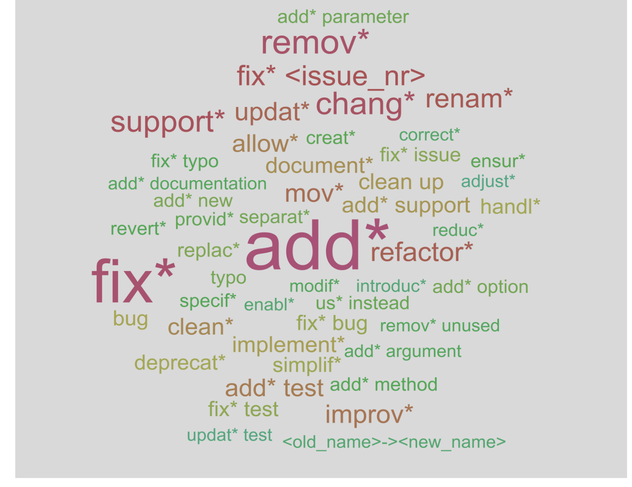


- Anschließender Vergleich der gefundenen Muster mit einer aktuellen Studie zu Java-Mustern [2] in dem Bereich.

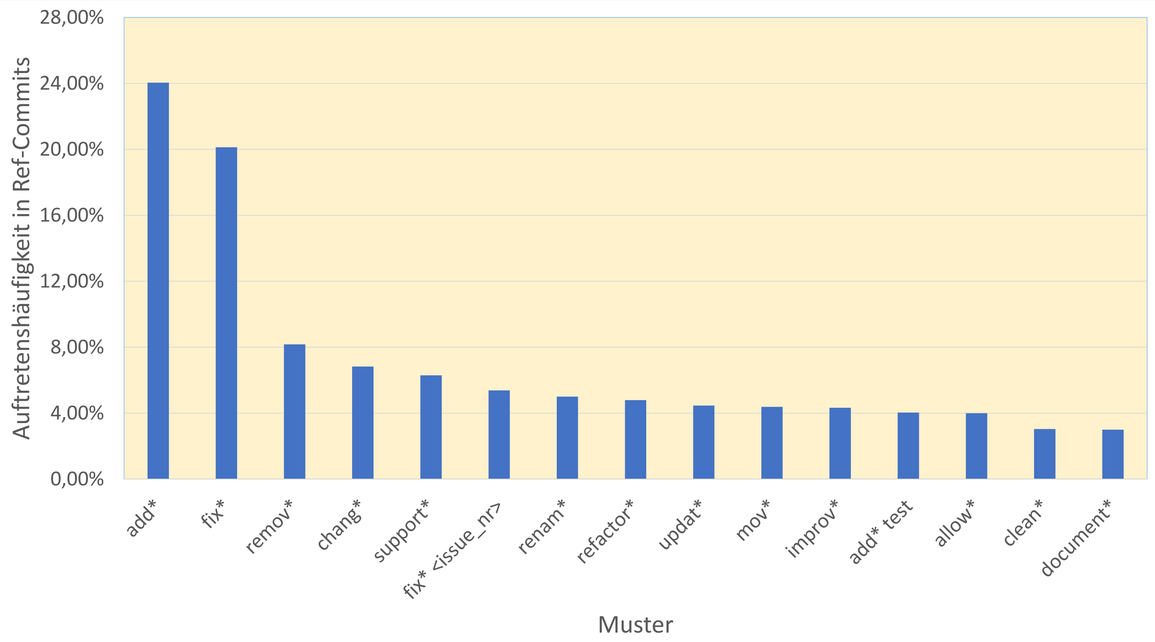
Ergebnisse zu Teil A

- Wir konnten 102 Muster feststellen, die in Refactoringcommits vermehrt auftraten.
- Unmittelbare Kommunikation von Refactorings durch den Begriff *refactor** (4,79% der Commits) oder der direkten Benennung von Refactoringtypen findet selten statt.
- Auftreten von Begriffen mit ähnlicher Semantik; z. B. *clean** und *simplif**
- Refactoringaktivität wird oft zusammen mit anderen Aktivitäten durchgeführt, welche dann kommuniziert werden.
  (bspw. *add** => Einführen von Features, *fix** => Beseitigung von Problemen und Bugs)

Die 50 häufigsten Muster



häufigste Muster Top-15

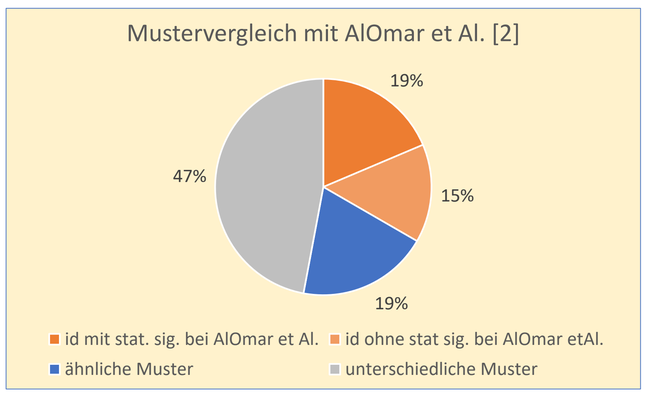


- Es existieren viele allgemeinere Muster, die mit Refactoringaktivität in Zusammenhang stehen können, aber auf unterschiedliche Weise verwendet werden, sodass sich nicht klar sagen lässt, ob sie im Zusammenhang mit Refactoringkommunikation relevant sind. Beispiele: *refin** und *rework**
- => Überprüfung der stat. Signifikanz des Auftretens dieser Muster in Refactoringcommits im Vergleich zu ihrem Auftreten in Nicht-Refactoringcommits in künftigen größer angelegten Studien notwendig.

Ergebnisse zu Teil B

Vergleich unserer Muster mit denen von AlOmar et Al. [2] für Java festgestellten Mustern:

- Wir konnten 34 nahezu identische Muster feststellen.
- 20 weitere Muster wiesen Ähnlichkeiten auf.
- Vor allem für die gemeinsam aufgetretenen Muster, die sich in AlOmar et Al.‘s Untersuchung als stat. signifikant herausstellten, vermuten wir eine hohe Relevanz im Hinblick auf Refactoring-kommunikation.



- 48 unterschiedliche Muster => mögliche Gründe: Unterschiede der verwendeten Refactoring-Detektionstools, ungleiche Kommunikation der Entwickler-Communities;
  Vermutung: Die betroffenen Muster sind eher allgemeiner Natur, weshalb unterschiedliche Muster für die Kommunikation gewählt wurden.
- *pep8* als python-spezifisches Muster festgestellt.

Fazit

- Stattgefundene Refactorings werden häufig nicht kommuniziert.
- Oft werden Refactorings auch im Rahmen einer anderen Aktivität durchgeführt, welche dann mitgeteilt wird.
- In den Refactoringcommits treten in Java und Python eine Reihe gemeinsamer wie auch ähnlicher Muster auf.
- Untersuchungen wie die unsrige können bei der (Fort-)Entwicklung von Refactoring-Dokumentations-tools helfen und einen Beitrag zu einer einheitlicheren Refactoringkommunikation leisten.

Literatur

1. Awesome python. https://github.com/vinta/awesome-python. Last accessed: June 15, 2022.
2. Eman Abdullah AlOmar, Anthony Peruma, Mohamed Wiem Mkaouer, Christian Newman, Ali Ouni, and Marouane Kessentini. How we refactor and how we document it? on the use of supervised machine learning algorithms to classify refactoring documentation. Expert Systems with Applications, 167:114176, 2021.
3. Hassan Atwi, Bin Lin, Nikolaos Tsantalis, Yutaro Kashiwa, Yasutaka Kamei, Naoyasu Ubayashi, Gabriele Bavota, and Michele Lanza. Pyref: Refactoring detection in python projects. In 2021 IEEE 21st International Working Conference on Source Code Analysis and Manipulation (SCAM), pages 136–141, 2021.
4. Yaroslav Golubev, Zarina Kurbatova, Eman Abdullah AlOmar, Timofey Bryksin, and Mohamed Wiem Mkaouer. One thousand and one stories: A large-scale survey of software refactoring. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, page 1303–1313, New York, NY, USA, 2021. Association for Computing Machinery.
5. Frieso Ritter. Kommunikation von refactorings in python-open-source-projekten. Bachelor’s thesis, Universität Hamburg, FB Informatik, 2022.