УДК 004.4

ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ ДЛЯ РЕФАКТОРИНГУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Сусла М.В.[1)], Папа О.А.[2)], Гладчук Є.З.[3)]
Західноукраїнський національний університет
[1)]викладач;[2)]к.т.н., старший викладач;[3)]магістрант

I. Постановка проблеми

Рефакторинг коду вирішує проблему архітектурного занепаду об'єктно-орієнтованих проектів програмного забезпечення, покращуючи їх внутрішню структуру без зміни поведінки. Рефакторинги покращують якість програмного забезпечення та його збереженість при правильному застосуванні. Однак виявлення можливостей для рефакторингу є складною задачею як для розробників, так і для дослідників. У з розвитком штучного інтелекту алгоритми машинного навчання показали великий потенціал для вирішення цієї проблеми.

II. Мета роботи

У цьому дослідженні метою є оптимізація RefactoringMiner для виявлення рефакторингів в відкритих проектах на Java та обчислено метрики коду за допомогою статичного аналізу. Ми визначили проблему виявлення можливостей для рефакторингу як задачу бінарної класифікації та використовували алгоритми машинного навчання для її вирішення. Моделі класифікують між конкретним типом рефакторингу та стабільним класом, використовуючи метрики як ознаки.

Рисунок 1–Візуальне представлення

III. Основна частина

Коротко кажучи, нам потрібен інструмент для збору даних, який обробляє всі репозиторії, об'єднуючи дані від Refactoring Miner, CK та інших метрик в рефакторинг чи стабільні екземпляри, і зберігає їх в базі даних. Процес збору даних подібний для кожного репозиторію, кожен репозиторій

обробляється індивідуально. Таким чином, процес збору може легко паралелізується для видобутку даних з кількох репозиторіїв одночасно. Зверніть увагу, що інструмент для збору даних може реєструвати багато деталей витягування даних, але з метою спрощення ця інформація в даному розділі не розглядається. Основні компоненти та етапи обробки інструменту для збору даних будуть пояснені далі.

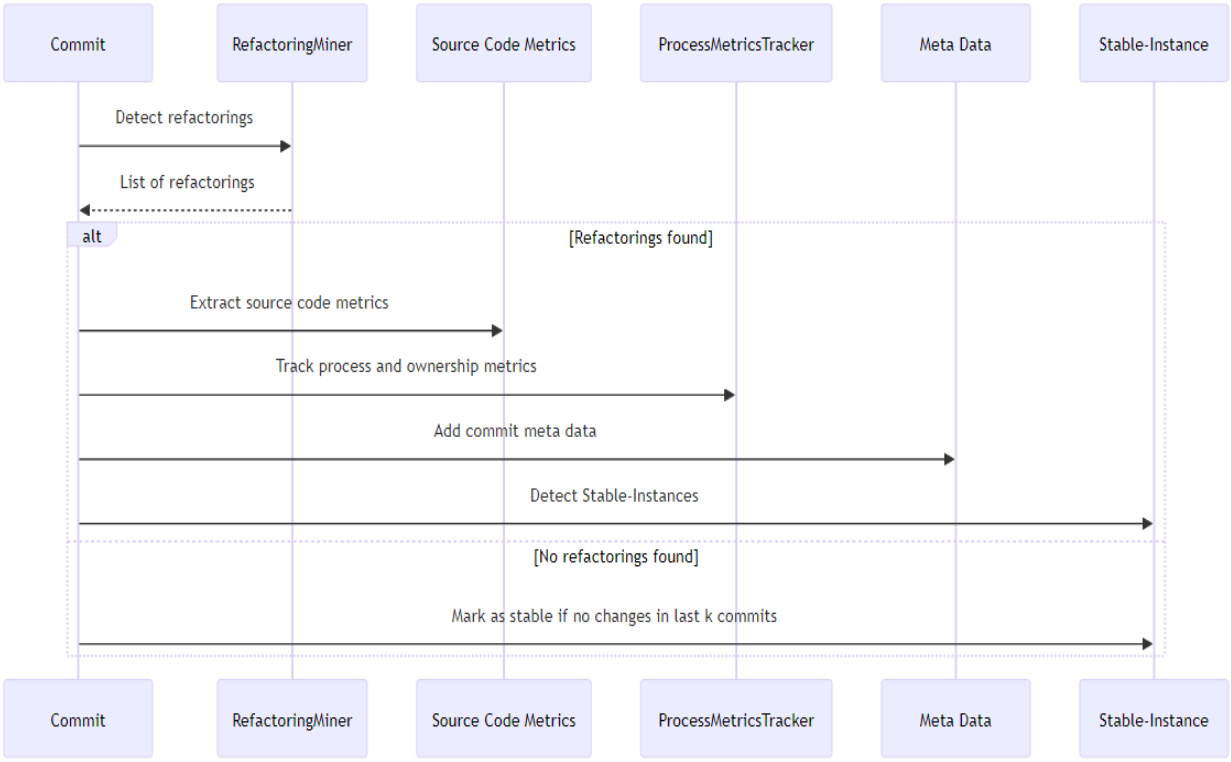

Рисунок 2–Послідовність дій починаючи з коміту

Коміт (C) надсилає запит до RefactoringMiner (RM) для виявлення рефакторингів. RefactoringMiner (RM) повертає список рефакторингів до Коміту (C).

Якщо рефакторинги знайдені:

- Коміт (C) витягує метрики вихідного коду за допомогою SourceCodeMetrics (SCM).
- Коміт (C) відстежує процес та метрики власності через ProcessMetricsTracker (PMT).
- Коміт (C) додає метадані коміту через MetaData (MD).
- Коміт (C) виявляє Stable-Instances (SI).

Якщо рефакторинги не знайдені:

- Коміт (C) позначає як стабільний, якщо не було змін у останніх k комітах через Stable-Instances (SI).

Висновок

Ця робота презентує, на нашу думку, досить великий набір даних рефакторингів відкритих проектів на Java на сьогоднішній день. У цьому наборі даних всі рефакторинги збагачені множиною метрик коду, процесу та власності, що описують рефакторинг на різних рівнях. Крім того, це дослідження зібрало нерефакторені класи (стабільні екземпляри), як від'ємні зразки для задачі прогнозування рефакторингів. Набір даних може бути використаний дослідниками для подальшого вивчення галузі рефакторингу програмного забезпечення та обслуговування вихідного коду.

Список використаних джерел

1. Jan Gerling and Mauricio Aniche. Appendix: Refactoring dataset, November2020. URL https://doi.org/10.5281/zenodo.4267711.
2. Fowler M. Refactoring: Improving the Design of Existing Code / MartinFowler., 2019. – 424 с.
3. Thainá Mariani and Silvia Regina Vergilio. A systematic review on search based refactoring. Information and Software Technology, 83:14–34, 2017.
4. Alberto S Nuñez-Varela, Héctor G Pérez-Gonzalez, Francisco E Martínez-Perez, and CarlosSoubervielle-Montalvo. Source code metrics: A systematic mapping study. Journal of Systems and Software, 128:164–197, 2017.