ACTIVE SUPPORT FOR CLONE REFACTORIZING: A PERSPECTIVE

Norihiro Yoshida (Nara Institute of Science and Technology)
Eunjong Choi (Osaka University)
Katsuro Inoue (Osaka University)
Outline

• Clone and its refactoring

• Previous research
  • Industrial experience in active clone refactoring

• Research proposal
  • Plan towards proactive support for clone refactoring

• Summary
What is Code Clone?

A code fragment that has similar or identical part in source code

Bug is found

Code Clone

Necessary to inspect them

If we modify one of them, it is necessary to determine whether or not we have to modify the others.
Clone Refactoring

Merging clones into a single program unit.
Clone Detection Tool: CCFinder [1]

- A token-based clone detection tool
  - Transform source code into sequence of tokens
  - Detect identical parts of the sequence as code clones
- Scalable and accurate

Industrial Experience in Clone Analysis

- Only hot clones can be refactored.
  - Strike while the iron is hot!

- In industry, cost of clone refactoring cannot be ignored.
  - Regression test after refactoring takes much cost to preserve behavior after refactoring.
  - Industrial developers do not touch source code after large-scale system test for releasing major version.

**Active support** of clone refactoring is needed to notify newly-appeared clones regularly to developers.

Applying to Industrial Process

- Applied Clone Notifier to development process in NEC
  - 6 programmers, 120 KLOC written in Java
- Got regular feedbacks from a project manager

10 sets of code clones were recognized as refactoring opportunities in 40 days.
Towards More Active Support

• Related Work: Proactive Support for Refactoring

• In the case of clone refactoring, existing tools lacks proactive support
  • There is no tool that supports clone refactoring based on the online analysis of code modification.

Proactive Refactoring Candidate Detection

Overview

- We plan to develop an Eclipse plug-in for proactive refactoring candidate detection.
  - The detection is based on the online analysis of code modification.
Proactive Candidate Detection for Clone Refactoring

Key Idea

When a developer refactors a code fragment, he/she has to consider to refactor the clones of the code fragment.

If IDE catches the beginning of clone refactoring, it can proactively recommend clones to be refactored at once.
Proactive Candidate Detection for Clone Refactoring

Step 1: Refactoring detection

Developer

Clone

Clone

Clone

Once he extracts a clone as a new method …

Tool detects the beginning of the refactoring

New method
Proactive Candidate Detection for Clone Refactoring

Step 2: Clone detection

Tool identifies the corresponding clones.
Proactive Candidate Detection for Clone Refactoring

Step3: Recommendation of refactoring candidates

Tool recommends to merge those clones into the new method.
Other Challenges for Proactive Support for Clone Refactoring

- Near real time clone detection
  - Invoked by modification in a source code editor
  - Finish the detection by the timing of next modification.

- Precise detection of refactoring candidate selection
  - Clones are not always refactoring candidates
    - Syntactically incomplete clone
    - Clone accidentally created by only the replacement of statements.
  - Discovering further heuristics is needed for eliminating clones that should not be detected.
Summary

• We discussed active and proactive support for clone refactoring
  • Industrial experience in active support for clone refactoring
  • Plan for developing tool that supports clone refactoring proactively
• Currently, we are developing a prototype tool that supports clone refactoring proactively.