Refactoring Planning for Design Smell Correction: Summary, Opportunities and Lessons Learned

Javier Pérez

September 24th 2013 - ICSM 2013 - Doctoral Symposium - Postdoctoral session
How do we refactor? Simple tools?

Only Simple Refactorings!
What refactorings do we use?


http://stackoverflow.com/questions/1804839/eclipse-most-useful-refactorings
How should we refactor? Recipes!

Disharmonies
chapters 5, 6, 7

Object-Oriented Metrics in Practice
Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object-Oriented Systems
Foreword by Stéphane Ducasse

“Bad Smells in Code” chapter 3

Cheatsheet “Smells to Refactorings”

Only textual descriptions!
Complex Refactoring Processes

- **Feature Envy:** `PrinterServer.formatSummary()`
- **Strategy:** move method from `PrintServer` to `Document`
- **Plan:**
  - rename `Document.formatSummary()`
  - move `PrintServer.formatSummary()`
Refactoring Strategies and Plans

“Refactoring Planning for Design Smell Correction in Object-Oriented Software”
PhD Thesis, July 2011, University of Valladolid, spain
Refactoring Strategies and Plans

“Refactoring Planning for Design Smell Correction in Object-Oriented Software”

PhD Thesis, July 2011, University of Valladolid, Spain
Hierarchical Task Network (HTN) Planning

- **Operators:** Low level changes - add / remove AST elements
- **Tasks:** Methods and operators - “recipes” about how to execute a task
- **Preconditions:** first-order-logic queries - gather system information
- **Goal:** Execute a task
Remove Data Class Plan

apply-refactoring:  show-method (org.jwebap.asm.attrs, stackmapattribute, gettypeinfolabels)
apply-refactoring:  move-method (from, org.jwebap.asm.attrs, stackmapattribute, getframelabels, to, org.jwebap.asm.attrs, stackmapframe, through, frame, keeping-delegate, false)
apply-refactoring:  show-method (org.jwebap.asm.attrs, stackmapattribute, writetypeinfo)
apply-refactoring:  move-method (from, org.jwebap.asm.attrs, stackmapattribute, writeframe, to, org.jwebap.asm.attrs, stackmapframe, through, frame, keeping-delegate, false)
apply-refactoring:  show-method (org.jwebap.asm.util.attrs, asmstackmapattribute, asmify)
apply-refactoring:  show-method (org.jwebap.asm.util.attrs, asmstackmapattribute, asmifytypeinfo)
apply-refactoring:  move-method (from, org.jwebap.asm.util.attrs, asmstackmapattribute, asmify, to, org.jwebap.asm.attrs, stackmapframe, through, f keeping-delegate, false)
apply-refactoring:  encapsulate-field (org.jwebap.asm.attrs, stackmapframe, label, getlabel, setlabel)
apply-refactoring:  encapsulate-field (org.jwebap.asm.attrs, stackmapframe, locals, getlocals, setlocals)
apply-refactoring:  encapsulate-field (org.jwebap.asm.attrs, stackmapframe, stack, getstack, setstack)
apply-refactoring:  remove-method(org.jwebap.asm.attrs, stackmapframe, setlabel) (SETTER)
apply-refactoring:  remove-method(org.jwebap.asm.attrs, stackmapframe, setlocals)(SETTER)
apply-refactoring:  remove-method(org.jwebap.asm.attrs, stackmapframe, setstack) (SETTER)
apply-remove-smell:  remove-data-class (org.jwebap.asm.attrs, stackmapframe) cleanclass
“Refactoring Planning for Design Smell Correction in Object-Oriented Software”

PhD Thesis, July 2011, University of Valladolid, spain
Opportunities and Future Work
Crowsourcing development knowledge: Mining Refactoring Strategies

Refactoring Planning Domain

✶ 3 Refactoring Strategies:
  ✶ for removing design smells
    › Remove Data Class and Remove Feature Envy
  ✶ for applying a complex refactoring
    › Move Method

✶ 9 Refactorings:
  ✶ Encapsulate Field, Move Method, Rename Method, Rename Field, Rename Parameter, Rename Local Variable, Remove Field, Remove Method and Remove Class.

✶ > 150 system queries:
Crowsourcing development knowledge: Mining Refactoring Strategies
Crowsourcing development knowledge: Mining Refactoring Strategies

Smell Finder → Refactoring Finder

S1, ... , Sn → R1, ... , Rn → Refactoring Strategy

Strategy Miner
Crowsourcing development knowledge: Mining Refactoring Strategies

Available Mature Tools

Our goal
Opportunities and Future Work

• Automated planning for other task scheduling problems in Software Reengineering

• Composite Refactoring Detection
  
  *Identifying composite refactorings with a scripting language; Huang et.al., 2011*

• Model Inconsistencies Removal

  *Resolving model inconsistencies using automated regression planning; Pinna-Puissant et.al., 2013*
Lessons Learned - technical

- There is a need to implement big refactorings
- Try not to automate too much
- Big refactorings should be implemented “empirically” - with recipes
- Reasoning should be done with the full AST information
Lessons Learned - general

- Invest enough time to find and define a realistic, relevant and feasible problem
- Ask the experts
- Research stays are a great investment
- Writing helps a lot settling things down
- Persevere!
Refactoring Planning for Design Smell Correction:
Summary, Opportunities and Lessons Learned

Javier Pérez
September 24th 2013 - ICSM 2013 - Doctoral Symposium - Postdoctoral session