Do Bad Smells Indicate “Trouble” in Code?

Min Zhang¹, Tracy Hall², Nathan Baddoo¹, Paul Wernick¹
¹University of Hertfordshire
²Brunel University
Introduction

- Background
- Systematic Literature Review
- Experiment Design
- Targeting Specific Bad Smells
Background

- Code Bad Smells (Fowler et al. 1999) are widely used for detecting software refactoring opportunities.

- However, no empirical evidence supports the efficacy of using Bad Smells.

- Fowler et al. (1999) are not specific about the problems caused by Bad Smells.
A Light-weight Systematic Literature review

- **Objective**: Summarise the current state of knowledge of Bad Smells.

- **Method**: Simplify the normal systematic literature review protocol:
  - Use only one database - IEEE Xplore.
  - Only consider 5 years 2002-2006.
  - Exclude the quality assessment process.
  - Involve only one researcher to extract research data.
Systematic Literature review Results

- Research attention varies between different Bad Smells.
  E.g. Duplicated Code attracted most attention.

- The status of knowledge varies between different Bad Smells.

- Previous studies of Bad Smells mainly use objective research data.

- Only a few empirical studies have been conducted to examine the effects of Bad Smells.
An empirical study of the impacts of Bad Smells

- Research Question:
  *What is the relationship between Bad Smells and faults?*

- Two Stage Study
  1. Using code-based metrics data.
  2. Using expert opinions.
Study using code-based metrics data

- Research Data: Eclipse

- Hypotheses:
  1. A class containing Bad Smells should correlate to faults.
  2. A class containing faults should correlate to Bad Smells.
Study using expert opinions

- Research Data: Expert opinions

- Research Method:
  - Conduct a survey of developers/researchers to capture their opinions about each Bad Smells’ relation with faults.

- Compare the results of this study with results of the first study.
Targeting Specific Bad Smells

Selection Criteria:

- Bad Smells which have attracted the least research attention in previous studies.

- Bad Smells which are relatively easy to identify using static source code analysis techniques.
  - The selected Bad Smell has relative formal definition.
  - Identifying selected Bad Smell needs no more than one version of source code.
### Table 4: Selected Code Bad Smells for Future Studies

<table>
<thead>
<tr>
<th>Bad Smell Name</th>
<th>Formal Definition</th>
<th>Can be Identified by using one version source code?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Divergent Change</td>
<td>2</td>
<td>N</td>
</tr>
<tr>
<td>Data Clumps</td>
<td>3</td>
<td>Y</td>
</tr>
<tr>
<td>Primitive Obsession</td>
<td>2</td>
<td>Y</td>
</tr>
<tr>
<td>Switch Statements</td>
<td>3</td>
<td>Y</td>
</tr>
<tr>
<td>Parallel Inheritance Hierarchies</td>
<td>3</td>
<td>N</td>
</tr>
<tr>
<td>Speculative Generality</td>
<td>3</td>
<td>Y</td>
</tr>
<tr>
<td>Message Chains</td>
<td>3</td>
<td>Y</td>
</tr>
<tr>
<td>Middle Man</td>
<td>3</td>
<td>Y</td>
</tr>
<tr>
<td>Inappropriate Intimacy</td>
<td>2</td>
<td>Y</td>
</tr>
<tr>
<td>Alternative Classes with Different interfaces</td>
<td>2</td>
<td>Y</td>
</tr>
<tr>
<td>Incomplete Library Class</td>
<td>1</td>
<td>N</td>
</tr>
<tr>
<td>Comments</td>
<td>1</td>
<td>Y</td>
</tr>
</tbody>
</table>
Thank you!

Q/A