Software Quality and Measurement

Prof.: Eduardo Figueiredo

Markos Viggiato de Almeida

Belo Horizonte

14 de setembro de 2016

When and Why Your Code Starts to Smell Bad

Michele Tufano, Fabio Palomba, Gabriele Bavota, Rocco Oliveto, Massimiliano di Penta, Andrea de Lucia, Denys Poshyvanyk

2015 IEEE/ACM 37th IEEE International Conference on Software Engineering

Agenda

- Introduction
- Goal
- Study Design
- Analysis of Results
- Threats to Validity
- Conclusion and Lessons Learned

Introduction

- Management of Technical Debt
- Two ways to add a functionality:

One is quick to do but is messy

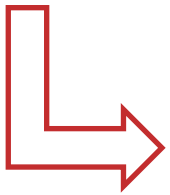
X

Cleaner design that will take longer

- Bad code smells

Bad Smells

- Relevance of code smells to developers
- Approaches and tools to detect them
- Side effects



Fault proneness, decrease of maintainability

What Do We Have Today?

- Tools don't consider circumstances
- Important to support developers in planning actions
- Common wisdom: maintenance activities and pressure to delivery the software

In This Paper...

- Large empirical study  200 projects
- Android, Apache and Eclipse
- Investigate WHEN code smells are introduced and WHY
- Analysis of code entities in change histories of projects

Large Study

- 0.5M commits were mined
- 9.164 of those identified as smell-introducing
- Results: quantitative/qualitative
- Often contradicting common wisdom

GOAL

- Analyze change history of projects with the purpose to find when and why smells are introduced

Study Design

Research Questions

- RQ1: When are code smells introduced?
  - To what extend common wisdom applies
  - Suddenly or gradually?
- RQ2: Why are code smells introduced?
  - Focus on the factors: commit goal, project status and developer status

Context Selection

- Change history of 200 projects hosted in Git repositories

Domain	#Projects	#Classes	KLOC	#Commits
Apache	100	4 - 5.052	1 – 1.031	207.997
Android	70	5 – 4.980	3 – 1.140	107.555
Eclipse	30	142 – 16.700	26 – 2.610	246.119
Overall	**200**	**-**	**-**	**579.671**

Domains

- Android: 70 open sources from f-droid
- Apache: 100 Java systems randomly selected from Apache website
- Eclipse: 30 randomly mined from GitHub

Domains

- Three main reasons:
  - Sizes: Android is smaller than the other ones
  - Architectures: Android mobile apps, Apache libraries and Eclipse plug-in
  - Developers team: Android has smaller teams

5 Types of Smells

- Blob Classes
- Class Data Should Be Private
- Complex Class
- Functional Decomposition
- Spaghetti Code

Data Extraction and Analysis

To Answer RQ1

- Analysis of repository r_i with tool *HistoryMiner* in order to find smell-introducing commits
- The tool checks each commit in chronological order and runs DECOR

DECOR

- High accuracy and efficient smell detector
- Rules based on internal quality metrics
- Our tool output: for each file f_j , the list of commits and if f_j was affected by a smell

Number of Commits

- For files affected by a smell: compute the number of commits until DECOR identify

- 2 scenarios:
  - One commit
  - After several commits

Number of Commits

- For 2nd scenario:
  - LOC, WMC, CBO, LCOM, NOA, NOM
- Compare their behavior in files based on a linear function

Metrics Comparison

- Linear function to approximate metrics distribution
- For files not affected: the first n commits (n = average)
- For affected files: from first commit until the commit identified by DECOR

To Answer RQ2

- Challenge: Identify the specific commit that introduced the smell
- Crucial to explain the reasons why the smell appeared

Smell-introducing Commit

- If it is the first commit 
- If after several commits 

- Results from RQ1 to select quality metrics

Smell-introducing Commit

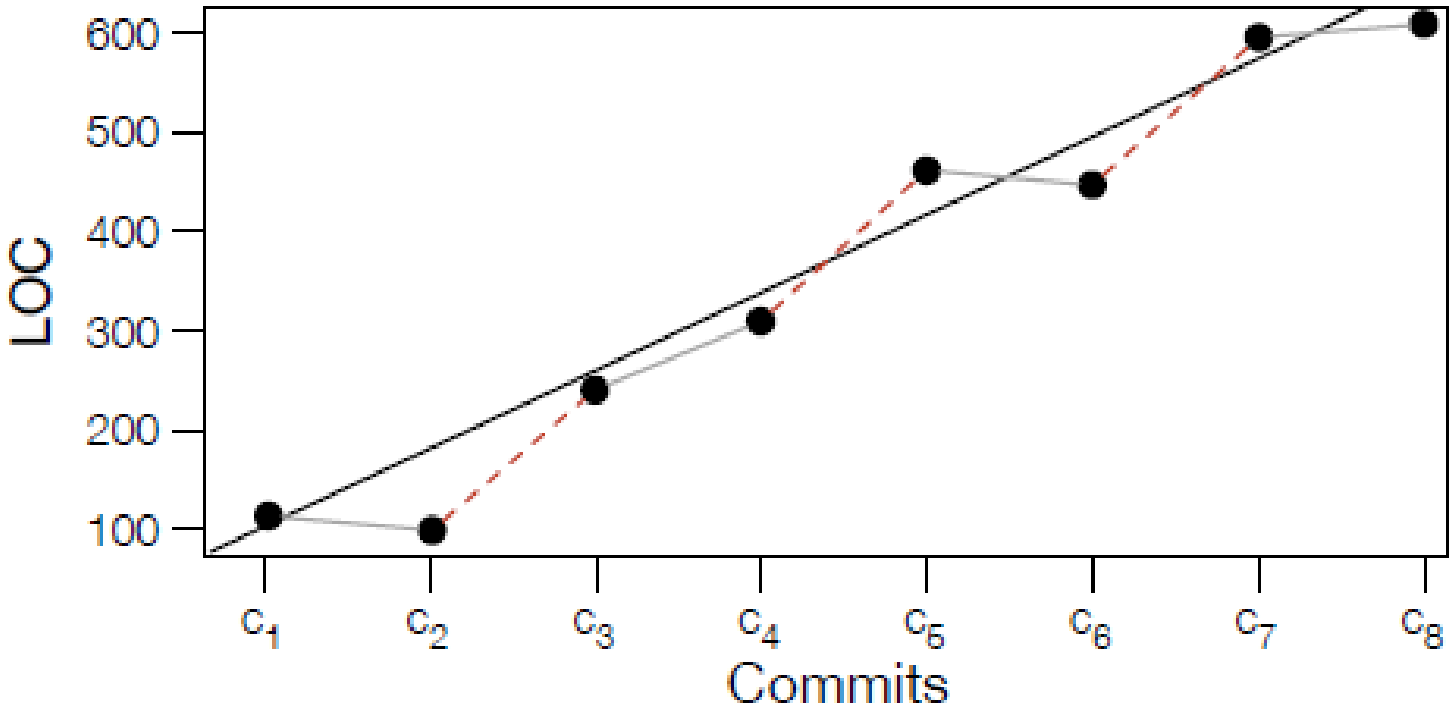


Fig. 1. Example of smell-introducing commit identification.

Smell-introducing Commit

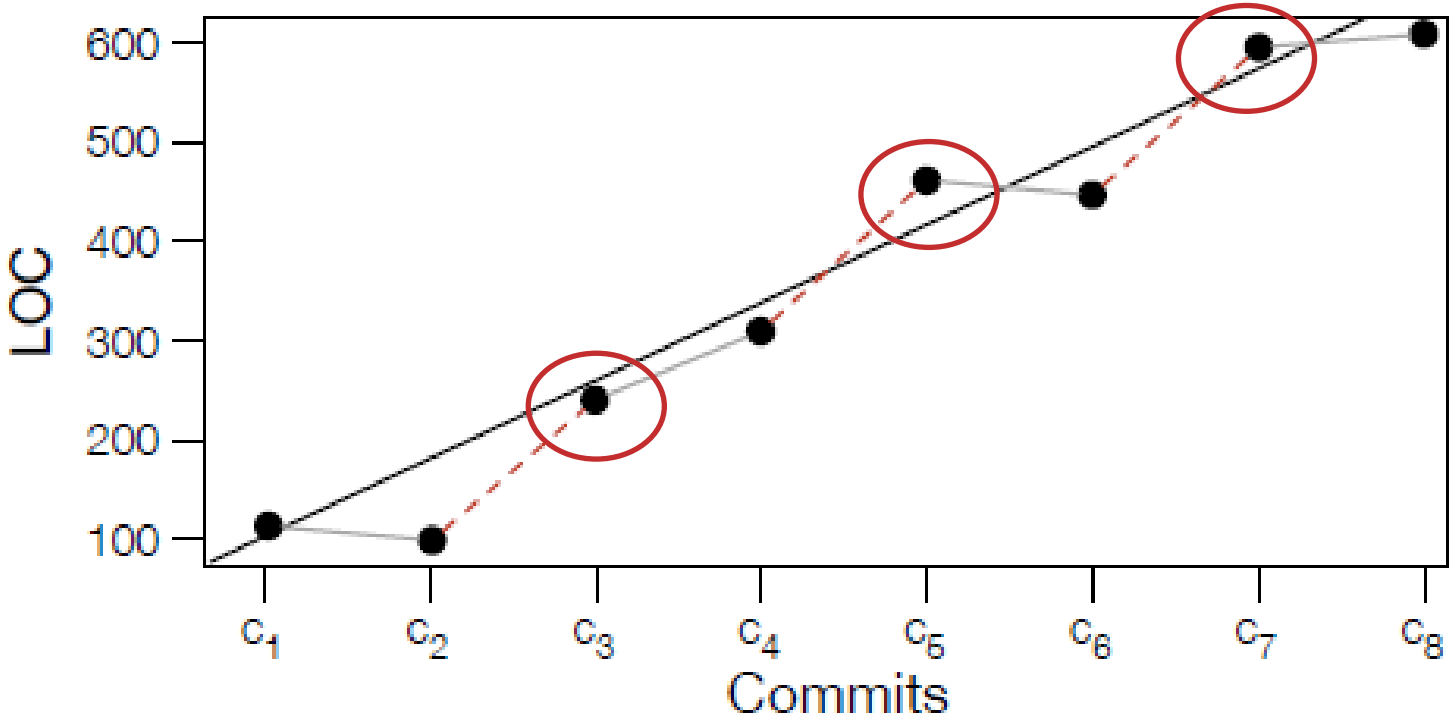


Fig. 1. Example of smell-introducing commit identification.

Smell-introducing Commit

- Overall, 9.164 smell-introducing commits in the 200 systems

Why smells were introduced?

- We assign tags to each commit

- Commit Goal: identify what the developer was doing

Bug Fixing, Enhancement,
New Feature, Refactoring

Why smells were introduced?

- Project Status: identify the stage of the project

Close to Release, Startup

Why smells were introduced?

- Developer Status: to identify how the developer was when smell was introduced

Workload, Ownership, Newcomer

Analysis of Results

- RQ1: Most of smells need one commit to manifest itself
- Contradicts the common wisdom

Analysis of Metrics

- For all smells, but Functional Decomposition, the affected files have higher slope

- When a smell is going to appear, the metric values increase very fastly

- LOC as main metric

Summary for RQ1

- Most of the smell instances are introduced when files are created
- Blob and Complex Class: smells manifest themselves after several changes performed on the file.
- Files that will become smelly exhibit specific trends for some quality metric values

Analysis of Results

- RQ2: analysis of percentage of smell-introducing commits in each tag

- Commit Goal
  - Enhancement operations (60% - 66%)
  - Refactoring (4% - 11%) $\longrightarrow$ quite unexpected

Project Status

- In the last month before release $\longrightarrow$ deadline pressure
- Considering project startup: many smells introduced a few months after start

Developer Status

- High workload influences negatively
- 55% of cases, high workload means introduction of smells
- Owner are responsible for most smells: against common wisdom

Summary for RQ2

- Smells are generally introduced when enhancing existing features or implementing new ones
- Last month before issuing a deadline
- Owners of the file are more prone to introduce smells

Threats to Validity

- Imprecision in measurements, but DECOR has high accuracy
- Cases of classes affected by many smells at the same time, but they represent 3% - 9% only

Conclusion and Lessons Learned

- This paper presented a large-scale empirical study to identify when and why smells are introduced
- Smells affect code since its creation → contradicts common wisdom

Conclusion and Lessons Learned

- Smells introduced when enhancing and add new features
- Owners are usually the responsible for introducing smells

OBRIGADO!