Software Quality and Measurement

Prof.: Eduardo Figueiredo

Giovani Romani Ferreguetti Junior

A Review-based Comparative Study of Bad Smell Detection Tools

Eduardo Fernandes, Johnatan Oliveira, Gustavo Vale, Thanis Paiva, Eduardo Figueiredo

Introduction

- Software maintenance and evolution are expensive activities
- Bad smell is a symptom
- Several tools have been proposed
- It's hard to:
  - Enumerate detection tools proposed
  - Say what bad smells they are able to detect
  - In which programming languages

Goal

- This study provides a systematic literature review(SLR)
- Comparative study with four available tools
- Asses: agreement, recall, precision, and usability

Systematic Literature Review

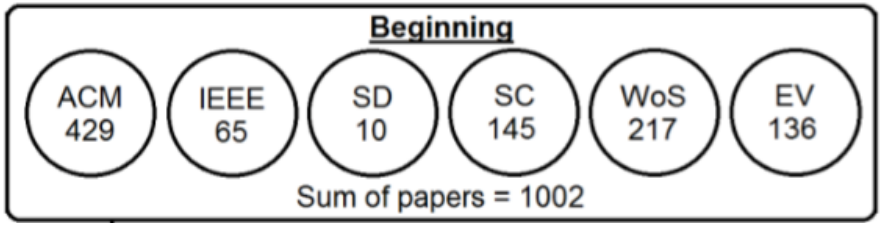
- Systematic Literature Review(SLR)
- Three steps:
  1. Planning
     - identify the need
     - research questions
  2. Conducting
     - selection of papers
     - data extraction
     - analysis
  3. Reporting

Research Questions

- Goal: identify and document all tools used for bad smell detection.
- RQ1: What are the bad smell detection tools proposed or used in literature papers?
- RQ2: Which are the main features of these tools?
- RQ3: Which are the most frequent types of bad smells these tools aim to detect?

Search String and Electronic Data Sources

- (tool* AND (“bad smell*” OR “design smell*” OR “code smell*” OR “design anomaly*” OR “code anomaly”))
- 6 electronic data sources:
  1. ACM Digital Library
  2. IEEE Xplore
  3. Science Direct, Scopus
  4. Web of Science
  5. Engineering Village.



Eletronic Datasources And Selection Criteria

- Refinement 1: remove duplicated studies.
- Refinement 2: remove non-papers.
- Refinement 3: inclusion/exclusion criteria.
- Refinement 4: do not use or propose bad smell detection tool.
- Refinement 5: look for citation of tools not covered.
- Not able to find papers that proposes tools.

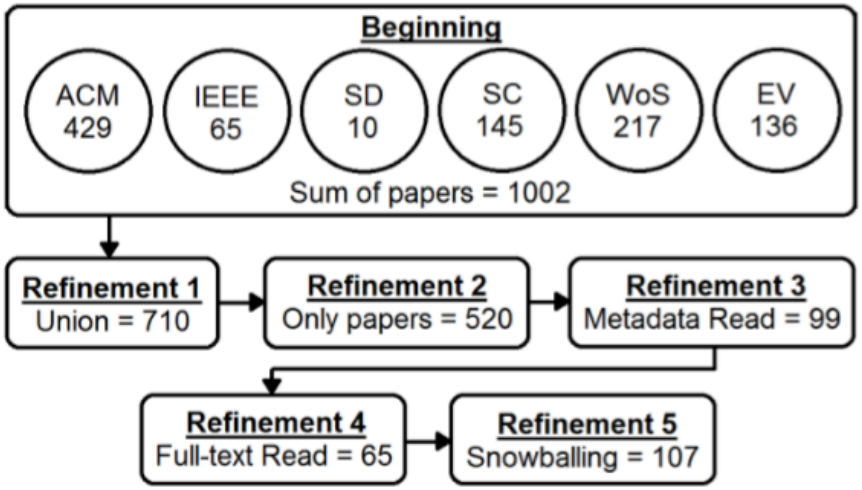


Figure 1. Steps for Selection of Papers

Table 1. List of SLR Inclusion and Exclusion Criteria

Inclusion Criteria	Exclusion Criteria
Papers published in Computer Science	Papers published before 2000
Papers written in English	Papers shorter than two pages
Papers available in electronic format	Websites, leaflets, and grey literature
Propose or use bad smell detect tools	

Data Extraction

- 84 bad smell detection tools
- Some tools do not have a specific section to describe features

Reporting

- Consider bad smells that shared a similar definition based on Fowler's definition.

Table 2. List of Bad Smells with Alternative Terms

Bad Smell	Alternative Terms
Duplicated Code	Code Clone, Clone Class, Clone Code, Cloned Code, Code Duplication, Duplicated Prerequisites
Large Class	Big Class, Blob, Brain Class, Complex Class, God Class
Long Method	Brain Method, God Method

I. SLR Results

- RQ1: What are the bad smell detection tools proposed or used in literature papers?

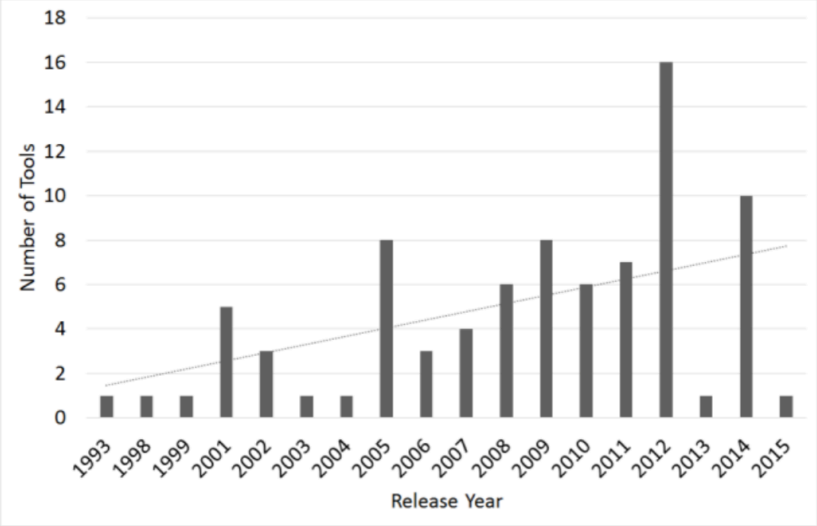


Figure 2. Number of Detection Tools Released by Year

Table 3. List of Bad Smell Detection Tools Found in this SLR

29 Tools Available Online for Download and Installation
Borland Together [77], CCFinder (CCFinderX) [29], Checkstyle [19], Clone Digger [8], Code Bad Smell Detector [22], Colligens [45], ConcernReCS [1], ConQAT [13], DECKARD [26], DuDe [75], Gendarme [53], inCode [77], inFusion [19], IntelliJ IDEA [17], iPlasma [43], Java Clone Detector (JCD) [28], jCosmo [71], JDeodorant [70], NiCad [10], NosePrints [53], PMD [19], PoSDef [9], SDMetrics [62], SpIRIT (JSpIRIT) [72], Stench Blossom [49], SYMake [67], TrueRefactor [20], Understand [65], Wrangler [37]
54 Tools Proposed in Literature but Unavailable Online
Absinthe [66], Anti-pattern Scanner [76], Arcoverde et al. [3], AutoMeD [78], Bad Smell Detection Tool (BSDT) [12], Bad Smells Finder [21], Bauhaus [59], Bayesian Detection Expert (BDTEX) [33], Bavota et al. [5], Baxter et al. [6], Bug Forecast [16], Clone Detector [64], CloneDetective [27], CocoViz [7], CodeSmellExplorer [57], CodeVizard [79], CP-Miner [38], Crespo et al. [11], Crocodile [63], DÉCOR [47], Dup [4], Duploc [14], EvoLens [58], Hamza et al. [23], Hayashi et al. [24], Hist-Inspect [42], iSPARQL [34], It's Your Code (IYC) [36], JCodeCanine [52], JSmell [61], Kaur and Singh [30], Keivanloo and Rilling [31], Kessentini et al. [32], Komondoor and Horwitz [35], Lui et al. [39], Matthew Munro [48], Mens et al. [46], Pradel et al. [56], PROblem DEtector O-O System (PRODEOOS) [44], Reclipse Tool Suite [73], Refactoring Browser [69], Ribeiro and Borba [60], SCOOP [40], Scorpio [25], Sextant [15], Smellchecker [55], SolidFX [68], Stasys Peldzius [54], SVMDetect [41], VCS-Analyzer [2], Wang et al. [74], WebScent [50], Xquery-based Analysis Framework (XAF) [51], Zang et al. [80]
1 Tool Cited but Unavailable Online for Download
Analyst [18]

II. SLR Results

- RQ2: Which are the main features of these tools?
  - Availability
    - 35.7% are plug-ins
    - 35.7% are standalone applications
    - 4.7% both plug-ins and standalone applications
    - 22.6% could not find information
  - Online documentation: 41.6%
  - Graphical User Interface(GUI): 71.4%
  - Detection Strategies:
    - 37% are metric-Based
    - 18% are based on trees
    - 7% are based on textual analysis
    - 6% are based on Program Dependency Graph
    - 3.5% based on token analysis
    - 20% no able to find.

III. SLR Results

- Did not find a bad smell detection tool for R
- Concentration in Java, C and C++

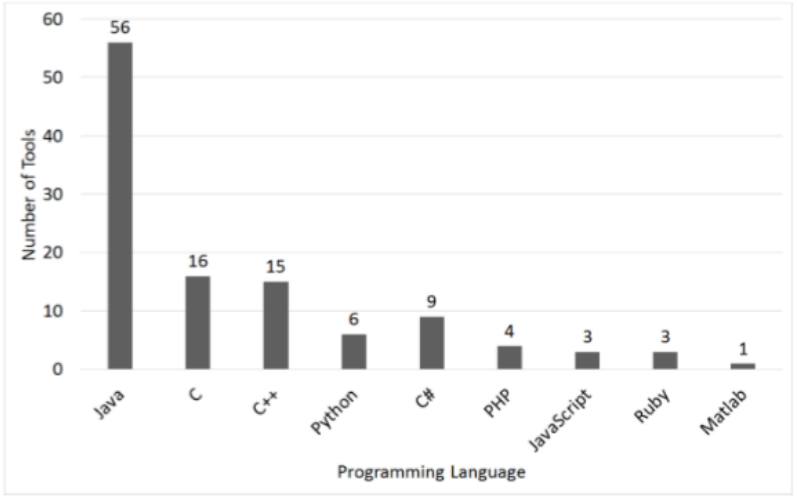


Figure 3. Programming Languages Tools Analyze

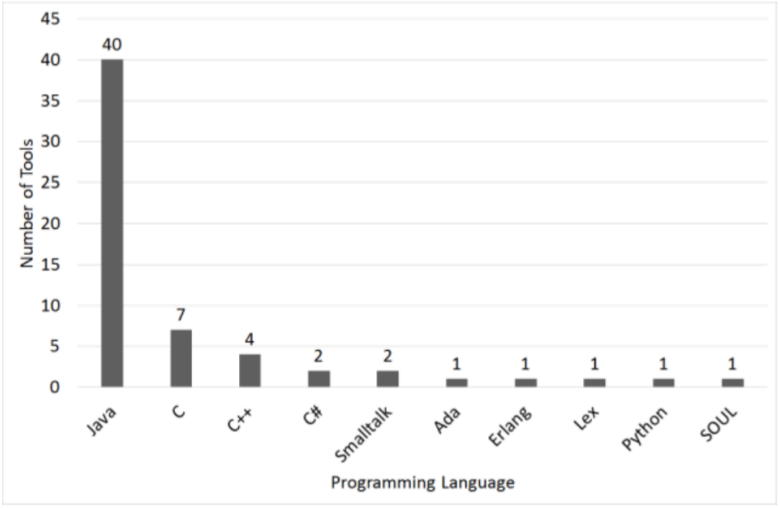


Figure 4. Programming Languages Tools Were Developed

IV. SLR Results

- RQ3: Which are the most frequent types of bad smells these tools aim to detect?
  - From 22 defined by Fowler, the found tools detect 20 bad smells of them
    - Alternative Classes with Different Interfaces
    - Incomplete Library Class
  - 41 defined by other authors
  - All ten most recurrent smells are from Fowler's book

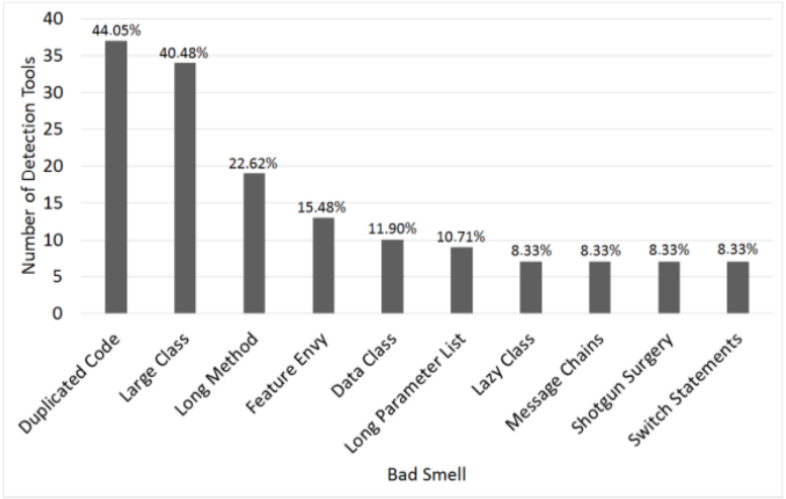


Figure 5. Top-ten Most Recurring Bad Smells

Comparative Study

- 29 bad smells detect tools able to download and install

Table 4. List of Bad Smell Detection Tools Available for Download

Tool Name	Plug-in	Detected Bad Smells	Language		Detection Technique	Free for Use	Guide	GUI	Release Year
			Developed	Detect					
Borland Together	Yes	Duplicated Code	Java	C#, C++, Java	Metrics	No	Yes	Yes	2011
CCFinder (CCFinderX)	No	Duplicated Code	C++	C, C#, C++, etc.	Token	Yes	Yes	Yes	2002
Checkstyle	Yes	Duplicated Code, Large Class, Long Method, Long Parameter List	Java	Java	NA	Yes	Yes	Yes	2001
Clone Digger	NA	Duplicated Code	Python	Java, Lua, Python	Tree	Yes	Yes	Yes	2008
Code Bad Smell Detector	No	Data Clumps, Switch Statements, and 3 other	Java	Java	AST	Yes	No	No	2014
Colligens	Yes	NA	C	C	NA	Yes	Yes	Yes	2014
ConcernReCS	Yes	Concern Smells: Primitive Concern Constant, and 5 other	Java	Java	Concern map	Yes	Yes	Yes	2012
ConQAT	Both	Clone Code	Java	ABAP, ADA, C++, C#, Java	Metrics	No	Yes	Yes	2005
DECKARD	No	Clone Code	C	Java	AST	Yes	Yes	No	2007
DuDe	No	Clone Code	Java	Language independent	Textual analysis	Yes	No	Yes	2005
Gendarme	No	Duplicated Code, Large Class, Long Method, and 4 other	C#	.NET, Mono	Rules	Yes	Yes	Yes	2006
inCode	Yes	Data Class, Data Clumps, Duplicated Code, and 2 other	Java	C, C++, Java	NA	No	Yes	Yes	2013
inFusion	No	Data Class, Data Clumps, Duplicated Code, and 2 other	NA	C, C++, Java	NA	No	Yes	Yes	2011
IntelliJ IDEA	No	Data Clumps, Feature Envy, Large Class, and 4 other	NA	Java, JavaScript, and 4 others	NA	No	Yes	Yes	2001
iPlasma	No	Duplicated Code, Feature Envy, Intensive Coupling, and 4 other	Java	C++, Java	Textual analysis	Yes	Yes	Yes	2005
Java Clone Detector	No	Duplicated Code	C++	Java	Tree	Yes	Yes	No	2009
jCosmo	No	InstanceOf, Switch Statement, Typecast	NA	Java	Tree	NA	Yes	Yes	2002
JDeodorant	Yes	Feature Envy, Large Class, Long Method	Java	Java	Metrics, AST	Yes	Yes	Yes	2007
JSpIRIT	Both	Data Class, Dispersed Coupling, Feature Envy, and 5 other	Java	C++, Java, Smalltalk	Metrics	Yes	Yes	Yes	2014
NiCad	Yes	Duplicated Code	C	C, C#, Java, etc.	NA	Yes	Yes	Yes	2011
NosePrints	Both	Feature Envy, Inappropriate Intimacy, Large Class, and 5 other	NA	NA	NA	No	No	Yes	2008
PMD	Both	Duplicated Code, Large Class, Long Method, Long Parameter List	Java	C, C#, C++, Java, PHP, and 11 other	NA	Yes	Yes	Yes	2008
PoSDef	Yes	NA	C#	UML diagrams	Metrics	Yes	No	Yes	2014
SDMetrics	No	Large Class	Java	UML diagrams	NA	No	Yes	Yes	2012
Stench Blossom	Yes	Comments, Data Clumps, and 4 other	Java	Java	Metrics	Yes	Yes	Yes	2010
SYMake	No	Cyclic Dependency, Duplicated Prerequisites	NA	C and Java	NA	Yes	Yes	Yes	2012
TrueRefactor	No	Lazy Class, Long Method, and 3 other	Java	Java	Graph	Yes	Yes	Yes	2011
Understand	No	NA	NA	C, C#, C++, etc.	NA	No	Yes	Yes	2008
Wrangler	Yes	Duplicated Code	Erlang	Erlang	Textual analysis	Yes	Yes	Yes	2010

Selection of Tools

- Selection process:
  1. Java programming language(most common)
  2. Sets of bad smells
  3. Free for use, at least in a trial version.
- 4 Tools remaining to be studied

Selection of Bad Smells

- Three most frequent bad smells detected
  - Duplicated code
  - Large class
  - Long method

- Duplicated code was discarded because it is hard to quantify the aimed results.

Selection of Applications

- Junit version 4
- MobileMedia version 9
- Recurrently used in previous quality and maintainability-related studies.
- Relied on two experts who used manual strategy for detecting, individually and manually, the bad smells in MobileMedia classes and methods.

Table 5. Size metrics of JUnit and MobileMedia

Size Metrics	JUnit	MobileMedia
Number of Classes	983	55
Number of Methods	2948	290
Lines of Code (LOC)	26456	3216

Table 6. Reference list of bad smells in MobileMedia

Bad Smell	Occurrences
Large Class	7
Long Method	6
Total	13

Overall Results

- Two personal computers with two different operating systems
- Two authors performed the same procedures
- Default tool setting
- Apart from JDeodorant tools have not detected Long Method instances in JUnit

Table 7. Bad Smell Detection in JUnit and MobileMedia

Tool Name	Junit		MobileMedia	
	LC	LM	LC	LM
inFusion	0	0	1	2
JDeodorant	88	48	11	12
PMD	12	0	1	3
JSpIRIT	6	0	2	5

Agreement, Recall and Precision

- The evaluated tools provided redundant detection results
- Recall and precision based on the bad smell reference list for the MobileMedia application

Table 8. Agreement of Tools for JUnit and MobileMedia

	Large Class			Long Method		
Application	**OA**	**AC1**	**95% CI**	**OA**	**AC1**	**95% CI**
JUnit	88.55%	0.87	[0.84, 0.90]	-	-	-
MobileMedia	88.79%	0.87	[0.79, 0.94]	97.27%	0.97	[0.96, 0.98]

Table 9. Recall and Precision of the Tools for MobileMedia Only

Bad Smell	**Recall**				**Precision**			
	inFusion	**JDeodorant**	**PMD**	**JSpIRIT**	**inFusion**	**JDeodorant**	**PMD**	**JSpIRIT**
Large Class	14%	14%	14%	14%	100%	9%	100%	50%
Long Method	33%	33%	50%	67%	100%	17%	100%	80%

Threats to Validity

- Scope and Strategy: other sources with relevant papers

- Validation and Generalization of Data and Results: cannot assume that all existing papers were included through the research string

Previous Work

- Fontana et al. Present a literature review and evaluation.
- Moha et al. Evaluates, but without providing a systematic literature review.
- None performs a systematic literature review and an extensive evaluation.

I. Conclusion and Lessons Learned

- Java, C and C++ are the top-three most covered programming languages
- Most of the 84 tools are implemented in Java and rely on metric-based detection technique
- Duplicated Code, Large Class and Long Method are the top-three bad smells the tools aims to detect
- Analyzed tools provide redundant detection results

II. Conclusion and Lessons Learned

- All tools presented some usability issues
- Usability evaluation:
  - Difficulty to navigate between bad smells occurrences
  - Difficulty to identify the source code related to a smell detection
  - Tools do not provide data visualization through statistical analysis

Table 10. Applicability Features of the Tools

Feature	inFusion	JDeodorant	PMD	JSpIRIT
Result export	X (in full version)	X		
Highlight smell occurrences	X	X	X	X
Allow detection settings	X	X	X	X
Graph visualization	X			
Detected Smell Filtering	X (in full version)	X	X	