A Review-based Comparative Study of Bad Smell Detection Tools

Um **estudo comparativo** baseado em **revisão de ferramentas de detecção de Bad Smell**

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

Software Engineering Laboratory (LabSoft)
DCC-UFMG

Apresentação: Maria Luísa Costa Pinto

Este trabalho apresenta uma revisão sistemática da literatura sobre ferramentas de detecção de bad smell. Além disso trazer um estudo comparativo de quatro ferramentas de detecção.

1. Introdução

Manutenção e evolução de software são atividades caras e podem representar até 75% dos custos de desenvolvimento de software.

Um bad smell é qualquer sintoma que possa indicar um problema de qualidade mais profundo no projeto ou no código do sistema.

Bad smells podem ser detectados no código-fonte usando análises manuais ou automatizadas. Existem muitas ferramentas de detecção de bad smell propostas na literatura, é difícil enumerá-las e dizer quais são os bad smells que elas são capazes de detectar.

1. Introdução

Trabalhos anteriores investigam o impacto de bad smells na qualidade de software e manutenção, estudando a detecção de bad smells no código.

- Fontana et al. apresentam uma revisão de literatura que abrange sete ferramentas de detecção de bad smell e avaliam quatro dessas ferramentas em termos de seus resultados de detecção.
- Moha et al. conduzem uma avaliação de uma ferramenta de detecção, chamada iPlasma, e a comparam com outras ferramentas.

1. Introdução

Nenhum desses estudos fornece uma visão abrangente de comparação de ferramentas de detecção de bad smell.

Este trabalho fornece uma **revisão sistemática da literatura (SLR) de ferramentas de detecção de bad smell**. Para cada ferramenta identificada, foram mostradas suas características, como sua linguagem de programação desenvolvida, linguagens compatíveis para detecção de bad smells, tipos suportados de bad smells, disponibilidade on-line para download, documentação disponível e ano de lançamento.

1. Introdução

Foi fornecida uma visão geral do **estado da arte em relação às ferramentas para detecção de bad smells**, com o objetivo de identificar tendências, desafios abertos e oportunidades de pesquisa.

Também foi realizado um estudo comparativo com quatro ferramentas disponíveis com relação à detecção de dois bad smells de Fowler: *Large Class* e *Long Method*.
A comparação de ferramentas visa avaliar a concordância (***Agreement***), a revocação (***Recall***), a precisão (***Precision***) e a usabilidade das ferramentas.

2. Revisão da literatura

Uma Revisão Sistemática da Literatura (SLR) fornece **identificação**, **análise** e **interpretação** de evidências apoiado por um protocolo. Três etapas compõem esse processo de SLR:

- **planejamento**: foi identificada a necessidade da revisão de literatura e, em seguida, definidas as questões de pesquisa e o protocolo de revisão
- **condução**: execução do protocolo definido, abrangendo desde a seleção inicial de artigos até a extração e análise de dados
- **geração de relatórios**: foram relatados os resultados obtidos

2.1 Objetivo e questões de pesquisa

O objetivo desta revisão sistemática da literatura é **identificar e documentar todas as ferramentas relatadas e usadas na literatura para detecção de bad smell.**

RQ1: Quais são as ferramentas de detecção de bad smell propostas ou usadas em artigos?
RQ2: Quais são as principais características dessas ferramentas?
RQ3: Quais são os tipos mais frequentes de bad smells que essas ferramentas visam detectar?

2.1 Objetivo e questões de pesquisa

Para responder a primeira questão de pesquisa, **RQ1 (quais as ferramentas)**, foi feita:

(i) uma busca automatizada em seis fontes de dados eletrônicas

(ii) uma filtragem manual para os resultados retornados.

Para o **RQ2 (quais as características)**, foram documentados os seguintes recursos de cada ferramenta: nome e ano de lançamento, tipo de disponibilidade, licença de usuário, linguagens de programação, malwares suportados e técnicas de detecção, disponibilidade de documentação e interface gráfica do usuário.

2.1 Objetivo e questões de pesquisa

Para responder a **RQ3 (quais bad smells)**, foram coletados os bad smells que os pesquisadores estão interessados, seja propondo ferramentas de detecção ou investigando-os em pesquisas.

2.2 Cadeia de pesquisa e critérios de seleção

Existem vários termos alternativos para o conceito de bad smell, como *design smell*, *code smell* e *code anomaly*.

```
(tool* AND ("bad smell*" OR "design smell*" OR "code smell*" OR "architecture smell*" OR "design anomaly*" OR "code anomaly*"))
```

A pesquisa foi aplicada apenas em metadados; ou seja, título, resumo e palavras-chave. Após a pesquisa do piloto, alguns termos gerais da string de pesquisa foram excluídos.

2.2 Cadeia de pesquisa e critérios de seleção

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	

2.2 Cadeia de pesquisa e critérios de seleção

Foram incluídos apenas artigos publicados após 2000 em nosso estudo por causa da publicação do livro Refactoring por Fowler em 1999, que define os bad smells mais conhecidos. As pesquisas-piloto também não retornaram nenhum trabalho relevante antes de 2000. Portanto, o estudo começou com a pesquisa de artigos publicados desde 2000 até 2015. No entanto, depois foram encontrados trabalhos publicados desde 1993.

2.3 Fontes de dados eletrônicas

Em julho de 2015, os autores executaram a cadeia de pesquisa definida em seis fontes de dados eletrônicas:

- ACM Digital Library
- IEEE Xplore
- Science Direct
- Scopus
- Web of Science
- Engineering Village

2.3 Fontes de dados eletrônicas

Inicialmente, começaram com 1002 artigos:

429 da ACM Digital Library (ACM)
65 da IEEE Xplore (IEEE)
10 da ScienceDirect (SD)
145 da Scopus (SC)
217 da Web of Science (WoS)
136 da Engineering Village (EV)

2.3 Fontes de dado eletrônicas

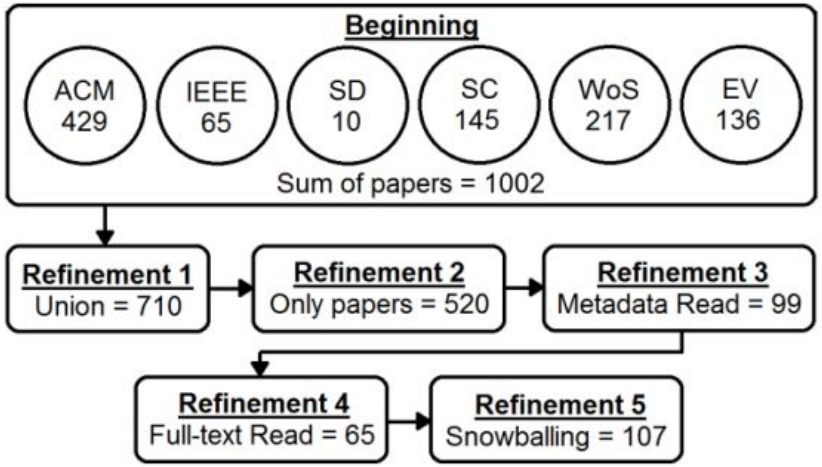

Figure 1. Steps for Selection of Papers

2.4 Extração de dados

Foi realizada uma leitura cuidadosa do texto completo dos 107 artigos selecionados. Com relação ao ano de lançamento das ferramentas, foi considerado o primeiro valor identificado.

Ao executar o protocolo SLR, 84 ferramentas de detecção de bad smell foram encontradas. Os dados extraídos sobre cada ferramenta foram salvos em uma planilha para análise.

2.5 Relatório

Como alguns bad smells têm a mesma definição, mas com nomes diferentes, foi necessário tratar esses casos para análise de dados.

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

3. Resultados

Ano de lançamento

das ferramentas

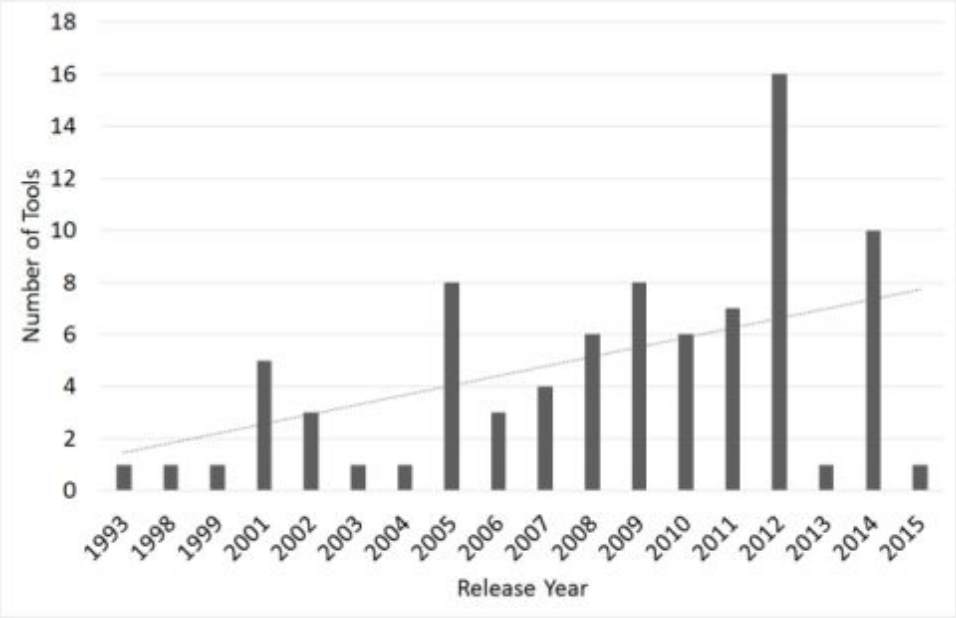

Figure 2. Number of Detection Tools Released by Year

3.2 Lista de ferramentas de detecção de bad smell

RQ1: Quais são as ferramentas de detecção de bad smell propostas ou usadas em documentos de literatura?

- 84 ferramentas de detecção de bad smell que foram encontradas na revisão da literatura.

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]

3.3 Principais Recursos das Ferramentas de Detecção

RQ2: Quais são as principais características dessas ferramentas?

35,7% das ferramentas (30 ferramentas) são plug-ins
35,7% (30 ferramentas) são aplicativos independentes

4,7% (4 ferramentas) estão disponíveis como plug-in e como aplicativo independente

22,6% (19 ferramentas) não continham informações sobre a disponibilidade

3.3 Principais Recursos das Ferramentas de Detecção

35 ferramentas de detecção de bad smell (41,6%) possuem documentação disponível on-line.

60 ferramentas (71,4%) fornecem uma interface gráfica de usuário (GUI) e apenas 4 ferramentas não fornecem GUI.

Para as 20 ferramentas restantes (23,8%), essas informações estavam indisponíveis.

3.3 Principais Recursos das Ferramentas de Detecção

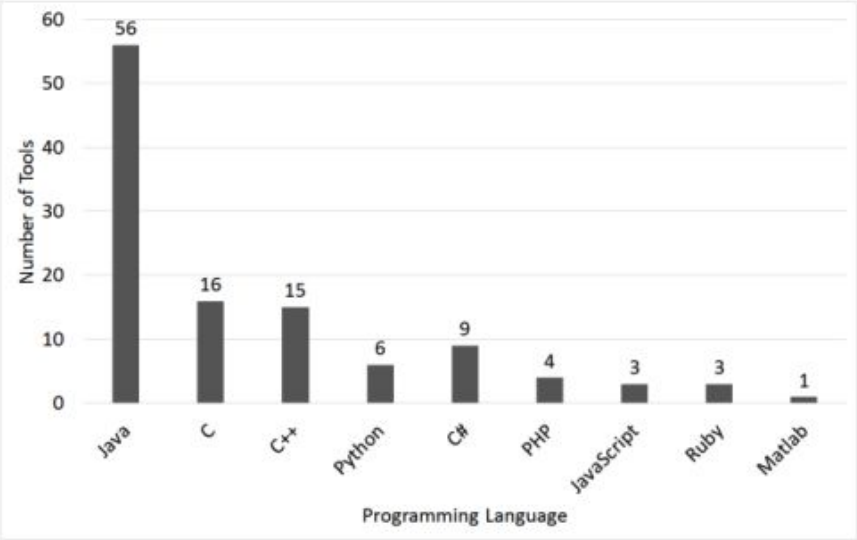

Figure 3. Programming Languages Tools Analyze

3.3 Principais Recursos das Ferramentas de Detecção

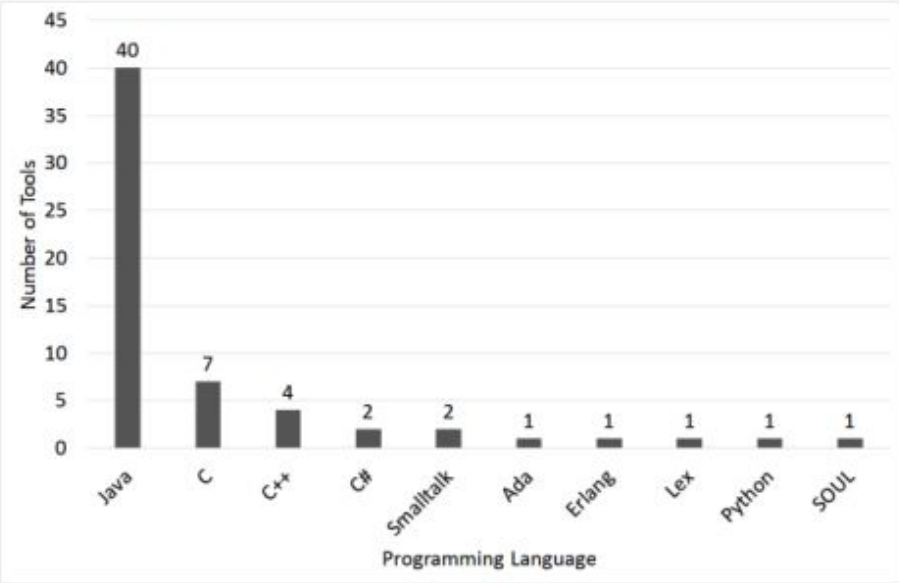

Figure 4. Programming Languages Tools Were Developed

3.4 Bad smells detectados

RQ3: Quais são os tipos mais frequentes de bad smells que essas ferramentas visam detectar?

Esta revisão encontrou **61 bad smells diferentes** que as ferramentas podem detectar. Dos 22 bad smells definidos por Fowler, as ferramentas encontradas visam detectar 20 delas.

41 bad smells definidos por outros autores são detectáveis pelas ferramentas.

3.4 Bad smells detectados

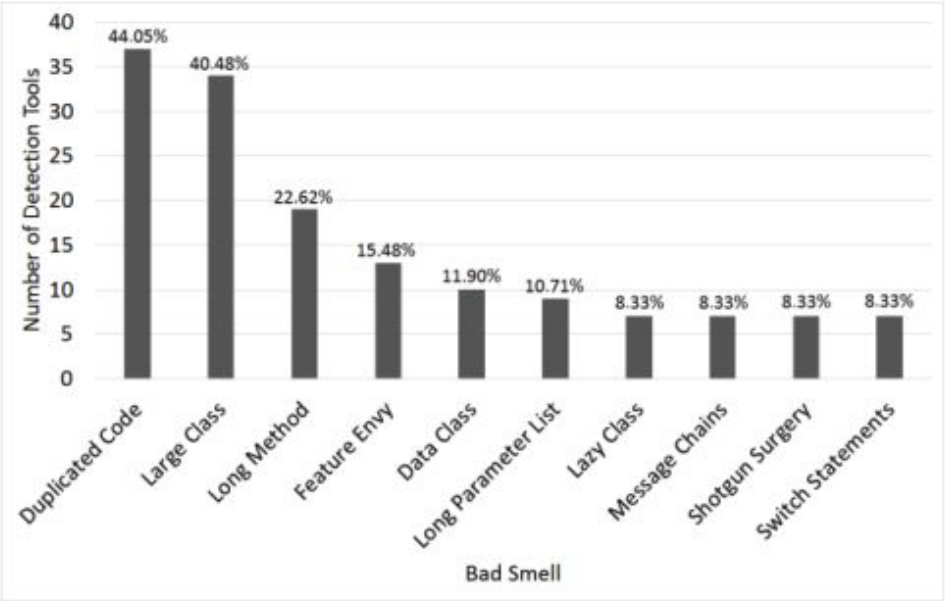

Figure 5. Top-ten Most Recurring Bad Smells

4. Um estudo comparativo

Esta seção apresenta uma tabela comparativa das **29 ferramentas** de detecção de bad smell que os autores puderam baixar e instalar.

Também foi feito um estudo comparativo detalhado de quatro ferramentas:

- InFusion
- JDeodorant
- PMD
- JSpIRIT

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

4.1 Seleção de ferramentas de detecção

O processo de seleção das 4 ferramentas analisadas foi:

- Escolher a linguagem de programação **Java**, escolher conjuntos de **bad smells**, restringir o conjunto de ferramentas para incluir **apenas ferramentas que estão livres para uso**.
- Depois de aplicar esses critérios, ficaram 8 ferramentas: Checkstyle, inFusion, iPlasma, JDeodorant, PMD, JSpIRIT, Stench Blossom e TrueRefactor.
- Checkstyle, iPlasma, TrueRefactor e Stench Blossom foram posteriormente descartados por diferentes razões.

4.2 Seleção de bad smells e Aplicações

Os três bad smells mais frequentes detectados pelas ferramentas são: *Duplicated Code*, ***Large Class***, e ***Long Method***. Código Duplicado foi descartado desta análise porque a natureza desse bad smell torna difícil quantificar os resultados desejados.

Software diferentes:

- **JUnit** (versão 4): é uma estrutura de teste Java de código-fonte aberto
- **MobileMedia** (versão 9): é uma linha de produtos de software (SPL) para aplicativos que manipulam foto, música e vídeo em dispositivos móveis.

Escolhemos estes dois sistemas de software porque eles foram usados recorrentemente em estudos anteriores relacionados à qualidade e manutenção.

4.2 Seleção de bad smells e Aplicações

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

4.3 Resultados Gerais

Para este estudo comparativo, foram configurados dois computadores pessoais com dois sistemas operacionais diferentes. Dois autores realizaram os mesmos procedimentos, cada um em um computador diferente.

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

4.4 Concordância (*Agreement*)

Para avaliar a concordância entre as ferramentas de detecção selecionadas, calculamos o coeficiente estatístico AC1. É um coeficiente de concordância robusto alternativo ao kappa de Cohen. Leva um valor entre 0 e 1 e relata o nível de concordância usando a seguinte escala:

- Fraco (<0,20),
- Regular (0,21 a 0,40),
- Moderado (0,41 a 0,60),
- Bom (0,61 a 0,80)
- Muito Bom (0,81 a 1,00).

4.4 Concordância

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]

JUnit

"Muito Bom" em relação ao *Large Class*.

MobileMedia

"Muito Bom" em relação ao *Large Class* e *Long Method*.

4.5 Revocação e Precisão

- **Precisão**: instâncias recuperadas que são relevantes
- **Revocação**: instâncias relevantes que são recuperadas

4.5 Revocação e Precisão

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%

Revocação: PMD e JSpIRIT forneceram os maiores resultados (*Long Method*).
Precisão: inFusion e PMD possuem os valores mais altos

5. Lições aprendidas

5.1 Código duplicado

Código Duplicado é o bad smell mais frequente que as ferramentas buscam detectar. É um bad smell complexo para detectar. Foi identificada uma oportunidade de pesquisa para que desenvolvedores de ferramentas trabalhem em como melhor apresentar os resultados desse bad smell de maneira mais fácil de ser quantificável e comparada.

5.2 Large Class e Long Method

No estudo comparativo de ferramentas com relação à detecção desses bad smells, concluímos que as ferramentas propostas fornecem resultados de detecção redundantes. Foi observada essa redundância pelos altos valores de concordância para as ferramentas estudadas que apontam para um acordo “Muito Bom”.

Também foi notado que o JDeodorant e o JSpIRIT contam com estratégias de detecção baseadas em métricas. Pode-se inferir que a redundância pode ser devido a técnicas de detecção semelhantes.

5.3 Avaliação de usabilidade

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	

6. Ameaças a validade

- **Escopo e Estratégia** - Para a revisão sistemática da literatura, foram selecionadas seis diferentes fontes de dados eletrônicos que agregam artigos de editoras diversificadas.

- **Validação e Generalização de Dados e Resultados** - Com relação ao protocolo de revisão, foi feita uma string de pesquisa para restringir nossa pesquisa. Esta cadeia inclui mais de 10 sinônimos de “bad smell” e, portanto, espera-se ter alcançado um número suficiente de artigos no contexto estudado.

6. Ameaças a validade

- **Search String Execution** - A string de busca foi executada em dois computadores diferentes, e foram baixados o BibTeX e os arquivos de texto três vezes, obtendo resultados idênticos.

- **Análise de texto completo e extração de dados** - Um dos autores foi responsável por ler integralmente os artigos selecionados e extrair deles ferramentas de detecção de bad smell propostas ou usadas, sem prazo para conclusão. Houve uma análise realizada por outro autor em 10% dos dados escolhidos aleatoriamente.

6. Ameaças a validade

- **Recursos de Catalogação das Ferramentas** - Não foi possível encontrar o nome de algumas ferramentas. Nestes casos, uma ferramenta foi nomeada com os autores que a propuseram, por exemplo.

- **A comparação das Ferramentas** - Ferramentas de Detecção tem como objetivo identificar o bad smell de diferentes maneiras, utilizando diversas técnicas e procedimentos. Além disso, algumas ferramentas não fornecem personalização de mecanismos de detecção, como limites para métricas. Portanto, todas as ferramentas avaliadas foram usadas em suas configurações padrão. Outras configurações provavelmente fornecerão resultados diferentes.

7. Trabalhos relacionados

Fontana et al. apresentam um estudo estreitamente relacionado. Eles discutiram as descobertas de uma **revisão de literatura (mas não sistemática)** abrangendo sete ferramentas de detecção, a saber, Checkstyle, DÉCOR, inFusion, iPlasma, JDeodorant, PMD e Stench Blossom. Além disso, eles avaliaram quatro ferramentas, **Checkstyle, inFusion, JDeodorant e PMD,** usando seis versões de um mesmo sistema de software como entrada.

Eles concluíram que as ferramentas fornecem resultados de detecção diferentes para um mesmo bad smell, e alguns resultados são redundantes. Eles encontraram resultados de concordância significativos apenas para dois bad smells: *Large Class* e *Long Parameter List*.

7. Trabalhos relacionados

Outro trabalho relacionado, de **Moha** et al. [47], avalia ferramentas de detecção, mas sem fornecer uma revisão sistemática da literatura. Os autores apresentam um estudo **comparativo de ferramentas**, incluindo uma nova proposta por eles, chamada de iPlasma.

Usando uma lista de bad smells criados através da inspeção manual do código-fonte, eles conseguiram calcular a revocação e a precisão para o iPlasma. No entanto, o estudo não compara um extenso conjunto de ferramentas de detecção e não há cálculo de concordância.

7. Trabalhos relacionados

Anteriormente, foi feita uma revisão de literatura de ferramentas de detecção de código duplicado. Neste estudo anterior, os autores investigaram as ferramentas disponíveis para projetos de software único no contexto da detecção de Duplicated Code entre projetos.

Neste artigo atual eles fornecem uma visão geral do **estado da arte em ferramentas de detecção de bad smell através de uma revisão sistemática da literatura**. Também foi apresentado um **estudo comparativo de ferramentas disponíveis** on-line para download e compatíveis com dois dos bad smells mais frequentes que as ferramentas visam detectar.

8. Conclusão

Bad smells são sintomas de anomalias no código-fonte que podem indicar problemas em um sistema de software. Foram encontrados um grande conjunto de **84 ferramentas diferentes, mas apenas 29 delas estão disponíveis on-line** para download.

Com relação às 84 ferramentas, **a quantidade de ferramentas autônomas e plug-in é praticamente a mesma**.
Java, C e C ++ são as três linguagens de programação mais cobertas para detecção de bad smell. A maioria das 84 ferramentas é implementada em Java e depende da técnica de detecção baseada em métrica. Por fim, a análise mostra que o ***Duplicated Code***, a ***Large Class*** e o ***Long Method*** são os três bad smells que as ferramentas visam detectar.

8. Conclusão

As ferramentas analisadas fornecem resultados de detecção redundantes, dado o alto coeficiente de concordância calculado.

Os resultados deste estudo comparativo indicam que o JDeodorant é a ferramenta que indica mais instâncias de bad smells em sua configuração padrão.
PMD obteve os resultados de detecção mais precisos para a *Large Class*, considerando as medições de recall e precisão. Com relação ao *Long Method*, as ferramentas PMD e JSpIRIT forneceram melhores resultados.

8. Conclusão

Além disso, este trabalho também contribuiu para avaliar alguns itens relacionados a usabilidade das ferramentas, que potencialmente servirão de referência aos desenvolvedores de ferramentas.