PPGCC
PROGRAMA DE PÓS-GRADUAÇÃO
EM CIÊNCIA DA COMPUTAÇÃO
DCC-UFMG

Estudo de Coocorrência de Padrões de Projeto e Bad Smells usando métricas de software

Bruno Luan de Sousa

Orientadora: Mariza Andrade da Silva Bigonha (UFMG)

Coorientadora: Kécia Aline Marques Ferreira (CEFET-MG)

07 de Julho de 2017

LLP Laboratório de Linguagens de Programação

Roteiro da Apresentação

- Contextualização
- Objetivo
- Revisão Sistemática da Literatura
- Metodologia
- *Design Pattern Smell*
- Estudo de Caso
- Ameaças à Validade
- Conclusão

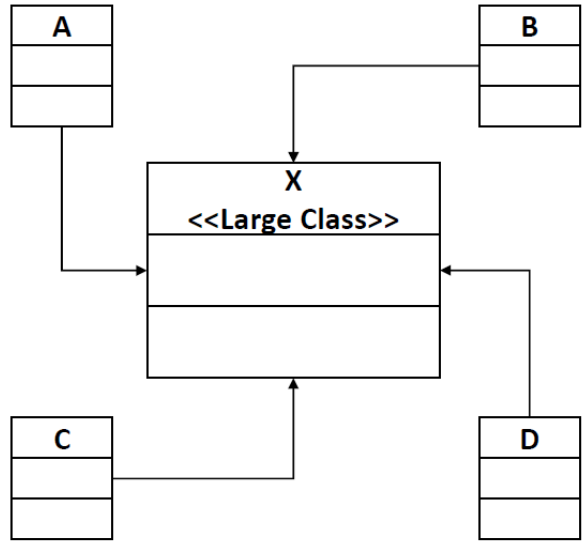
Contextualização

Padrões de Projeto

- Soluções gerais para problemas recorrentes
- Auxiliam na criação de sistemas com qualidade
- Boas práticas de programação
- Catálogo *GOF* são os mais conhecidos

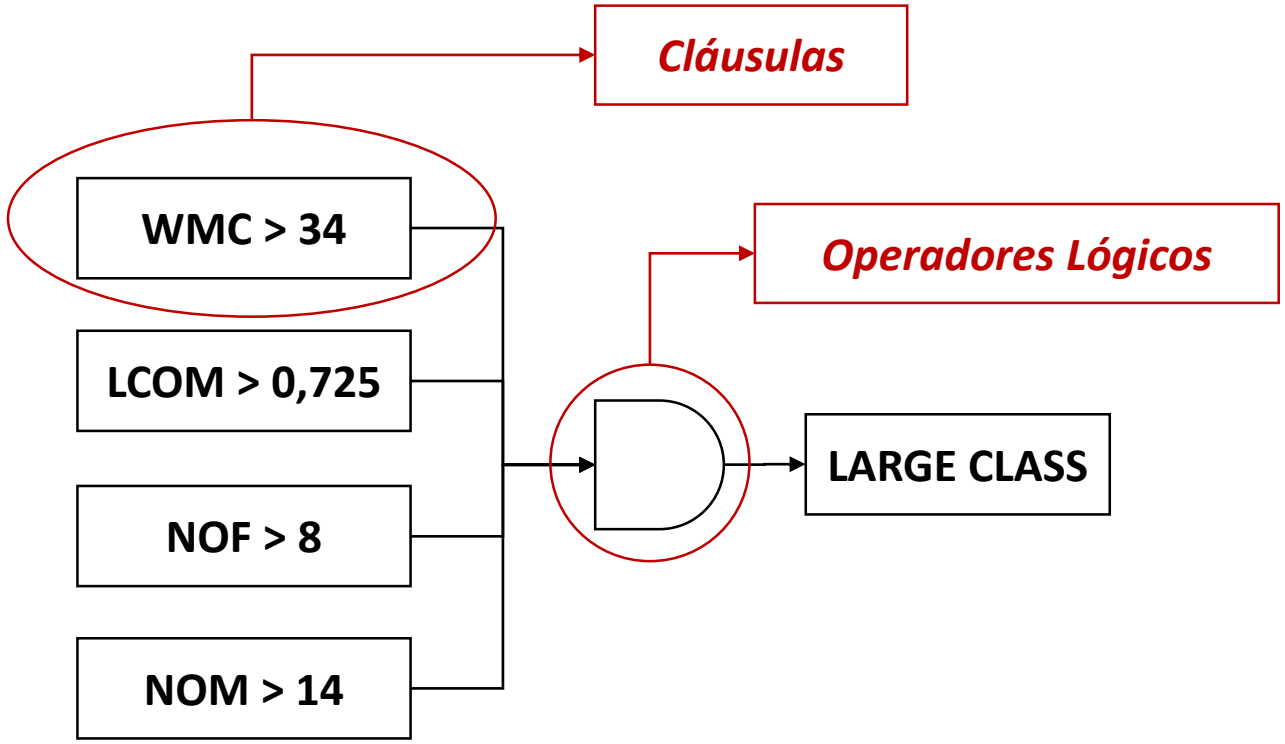
Bad Smells

- Sintomas de problemas no código fonte ou estrutura de um software
- Afetam negativamente a qualidade de um software
- Necessidade de Refatoração
- Detecção de *Bad Smells*
  - Inspeção manual
  - Estratégias de Detecção



Bad Smells

- Estratégias de Detecção
  - Regras formas que caracterizam um *bad smell* específico.



Métricas de Software

- Valor numérico relacionado a alguma dimensão do software
  - Exemplo: número de métodos

- Uso de métricas
  - Avaliação da qualidade um software
  - Identificação de desvios de código

- Valores referência são essenciais

Valores Referência

- Caracterizam uma métrica de software
  - Exemplos: bom, ruim, alto, baixo, etc.

- Valores Referência
  - Auxiliam métricas na avaliação de qualidade de um software
  - Trabalhos tem disponibilizado catálogos com valores referência
  - Filó (2014) apresenta maior quantidade de valores referência para métricas

- Podem ser utilizados para detecção de *bad smells*

Objetivo

Objetivo do Estudo

- Objetivo Geral
  - Investigar e estudar coocorrências entre padrão de projeto e *bad smells*
  - Sistemas de software orientado por objetos

- Objetivos Específicos
  - Verificar se padrões de projeto evitam ocorrências de bad smells
  - Identificar coocorrências
  - Detectar situações que contribuíram para o surgimento de coocorrências

Objetivo do Estudo

Bad Smell	**Padrões de Projeto**		
Data Class	*Adapter*	*Bridge*	*Command*
Feature Envy	*Composite*	*Decorator*	*Factory Method*
Large Class	*Observer*	*Prototype*	*Proxy*
Long Method	*State*	*Strategy*	*Singleton*
Refused Bequest	*Template Method*	*Visitor*	

Questões de Pesquisa

QP1. Os padrões de projeto definidos no catálogo *GOF* evitam a ocorrência de *bad smells* em software?

QP2. Quais padrões de projeto do catálogo *GOF* apresentaram coocorrência com *bad smells*?

QP3. Quais são as situações mais comuns em que *bad smells* aparecem em sistemas de software que aplicam os padrões de projeto *GOF*?

Revisão Sistemática da Literatura

Revisão Sistemática

- Por quê?
  - Resumir um tema, tecnologia ou área abordada
  - Extrair novos desdobramentos

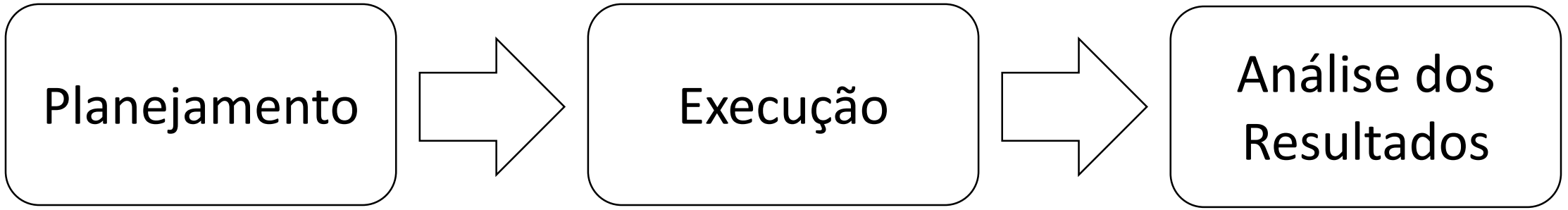
- Objetiva-se coletar evidências da relação entre padrões de projeto e *bad smells*

- Fornecer um *background* e fundamentar esse trabalho

Modelo de Revisão Sistemática

- Formulação de questões de pesquisa
- Seleção de fontes
- Seleção dos estudos
- Extração de informação
- Sumarização dos resultados

Processo da Revisão Sistemática

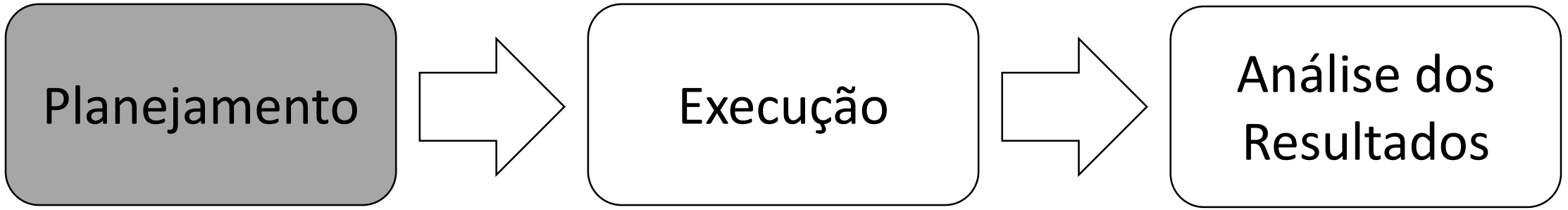


- Questões de pesquisa
- *String* de busca
- Fontes de pesquisa
- Critérios de inclusão e exclusão

- Processo de busca
- Processo de seleção dos estudos
- Sumarização dos resultados

- Análise dos estudos obtidos
- Discussão dos resultados

Processo da Revisão Sistemática



- Questões de pesquisa
- *String* de busca
- Fontes de pesquisa
- Critérios de inclusão e exclusão

- Processo de busca
- Processo de seleção dos estudos
- Sumarização dos resultados

- Análise dos estudos obtidos
- Discussão dos resultados

Questões de Pesquisa

QP1(RSL). Como a literatura tem abordado a relação entre padrões de projeto e *bad smells*?

QP2(RSL). A literatura tem explorado coocorrências entre padrões de projeto e *bad smells*?

Questões de Pesquisa

QP2.1(RSL). Quais bad smells são abordados pela literatura na identificação de coocorrências com padrões de projeto?

QP2.2(RSL). Quais padrões de projeto são usados pela literatura na identificação de coocorrências com *bad smells*?

String de Busca

(“code smell” OR “code smells” OR “bad smell” OR “bad smells” OR anti pattern” OR antipatterns” OR “anti-pattern”) AND (“design patterns” OR “design pattern”)

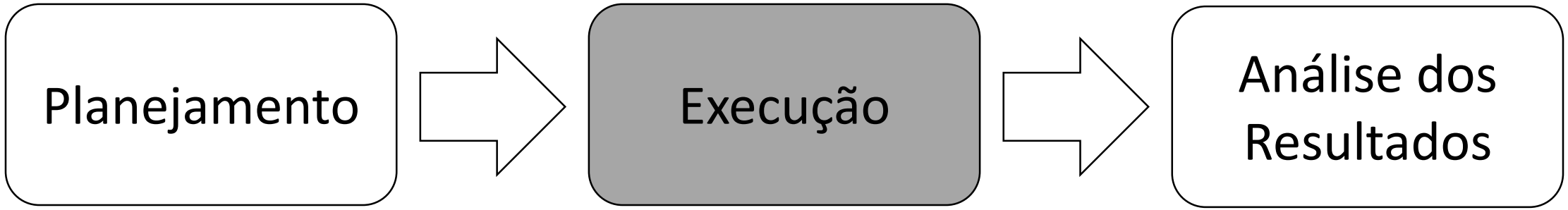
Fontes de Pesquisa

#	Base de Dados	Endereço
1	ACM Digital Library	http://dl.acm.org/
2	Compendex (Engineering Village)	https://www.engineeringvillage.com
3	IEEE	http://ieeexplore.ieee.org/
4	Science Direct	http://www.sciencedirect.com/
5	Scopus	http://scopus.com/
6	Springer	http://link.springer.com/
7	Web of Science	http://webofknowledge.com/

Critérios de Inclusão e Exclusão

Critérios de Inclusão
Documentos publicados em inglês
Documentos completos
Documentos publicados em Ciência da Computação
Documentos disponíveis em formato eletrônico
Documentos publicados em conferências e revistas
Documentos relacionados aos termos da *string* de busca
Critérios de Exclusão
Documentos classificados como tutoriais, pôsteres, painéis, palestras, mesas redondas, oficinas, teses, dissertações, capítulos de livro e relatório técnico
Documentos duplicados
Documentos que não podem ser localizados

Processo da Revisão Sistemática



- Questões de pesquisa
- *String* de busca
- Fontes de pesquisa
- Critérios de inclusão e exclusão

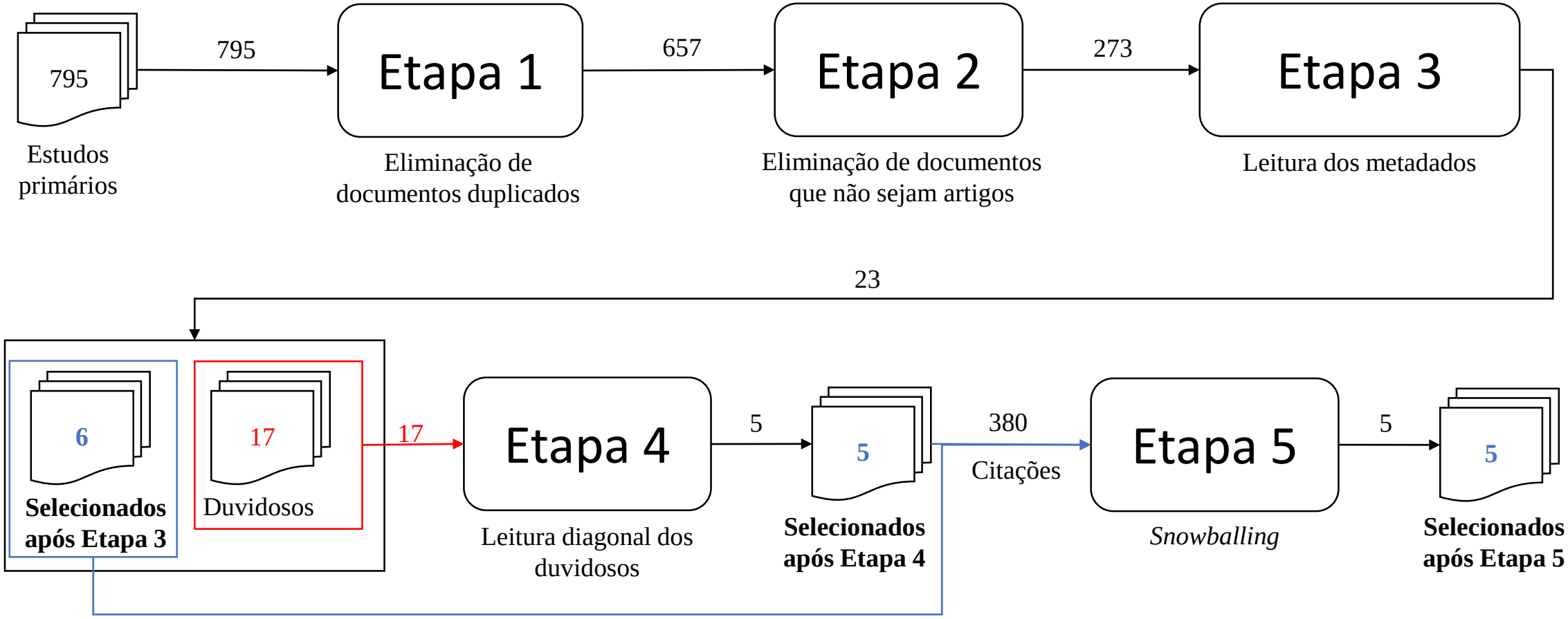
- Processo de busca
- Processo de seleção dos estudos
- Sumarização dos resultados

- Análise dos estudos obtidos
- Discussão dos resultados

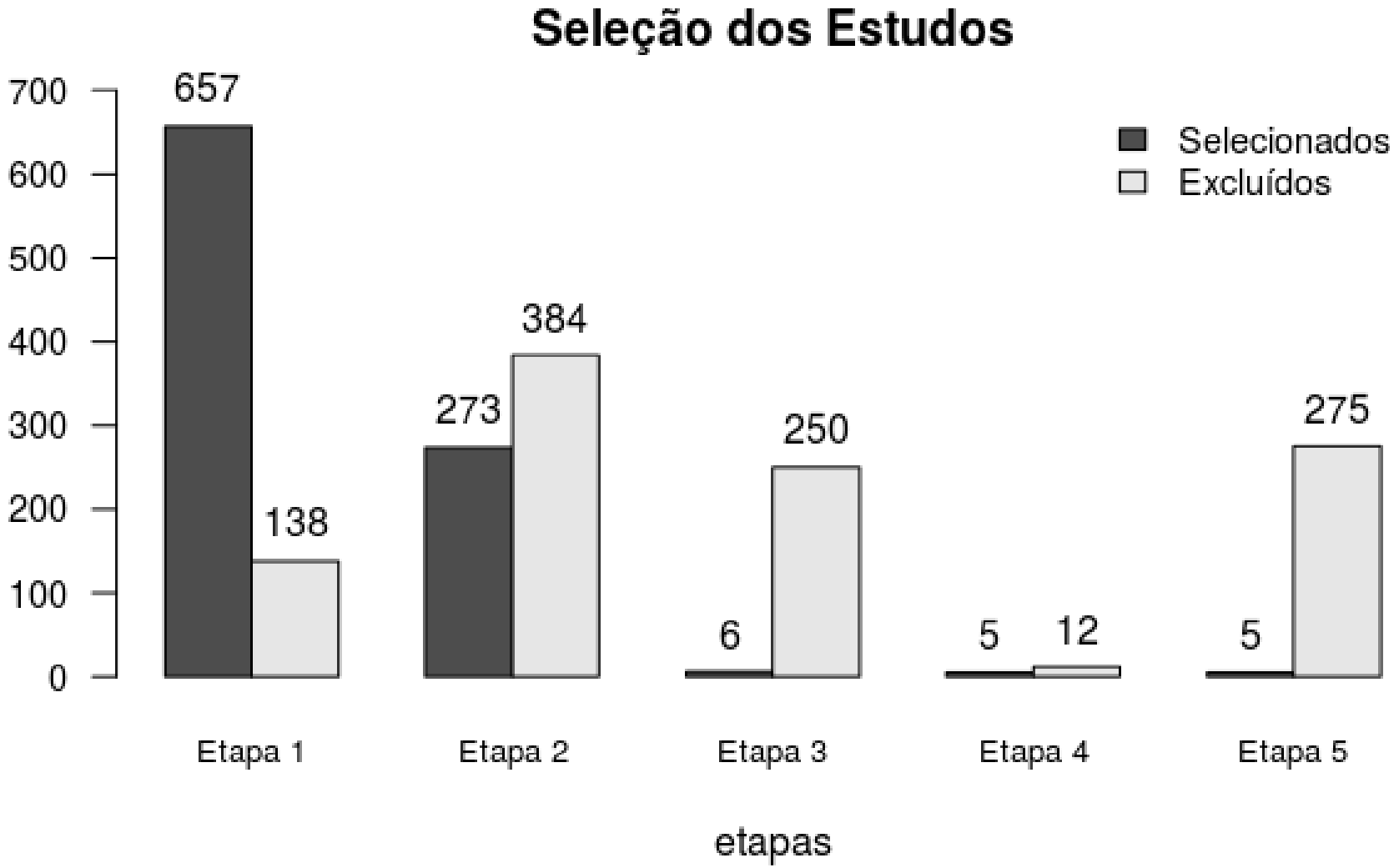
Processo de Busca

#	Base de Dados	Estudos Retornados
1	ACM Digital Library	12
2	Compendex (Engineering Village)	57
3	IEEE	0
4	Science Direct	176
5	Scopus	86
6	Springer	433
7	Web of Science	31
Total		**795**

Processo de Seleção dos Estudos



Processo de Seleção dos Estudos



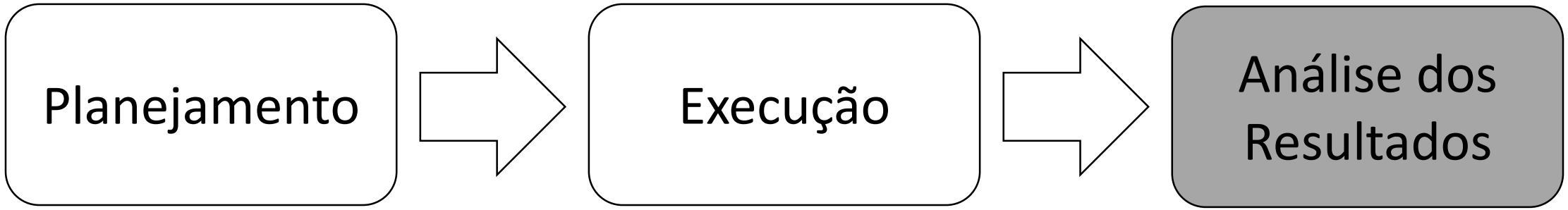
Sumarização dos Resultados

#	Título	Autor	Ano	Relação
1	Assessment of Design Patterns During Software Reengineering: Lessons Learned from a Large Commercial Project	Wendorff	2001	Impacto em qualidade de software
2	Coupling of Design Patterns: Common Practices and Their Benefits	McNatt & Bieman	2001	Impacto em qualidade de software
3	Defect frequency and design patterns: An empirical study of industrial code	Vokac	2004	Impacto em qualidade de software
4	Do Design Patterns Impact Software Quality Positively?	Khomh & Gueheneuce	2008	Impacto em qualidade de software
5	Automated refactoring to the Strategy design pattern	Christopoulou et al.	2012	Refatoração
6	Analysing Anti-patterns Static Relationships with Design Patterns	Jaafar et al.	2013	Coocorrência
7	A multiple case study of design pattern decay, grime, and rot in evolving software systems	Izurieta & Bieman	2013	Impacto em qualidade de software
8	Code Quality Cultivation	Speicher	2013	Impacto em qualidade de software

Sumarização dos Resultados

#	Título	Autor	Ano	Relação
9	Automated pattern-directed refactoring for complex conditional statements	Liu et al.	2014	Refatoração
10	Automatic recommendation of software design patterns using anti-patterns in the design phase: A case study on abstract factory	Nahar & Sakib	2015	Refatoração
11	A proposal of software maintainability model using code smell measurement	Wagey et al.	2015	Impacto em qualidade de software
12	Co-Occurrence of Design Patterns and Bad Smells in Software Systems: An Exploratory Study	Cardoso & Figueiredo	2015	Coocorrência
13	ACDPR: A Recommendation System for the Creational Design Patterns Using Anti-patterns	Nahar & Sakib	2016	Refatoração
14	Evaluating the impact of design pattern and anti-pattern dependencies on changes and faults	Jaafar et al.	2016	Coocorrência
15	The relationship between design patterns and code smells: An exploratory study	Walter & Alkhaeir	2016	Coocorrência
16	Automated refactoring of super-class method invocations to the Template Method design pattern	Zafeiris et al.	2017	Impacto em qualidade de software

Processo da Revisão Sistemática



- Questões de pesquisa
- *String* de busca
- Fontes de pesquisa
- Critérios de inclusão e exclusão

- Processo de busca
- Processo de seleção dos estudos
- Sumarização dos resultados

- Análise dos estudos obtidos
- Discussão dos resultados

Padrões de Projeto x Bad Smell

QP1(RSL). Como a literatura tem abordado a relação entre padrões de projeto e *bad smells*?

Relação	Quantidade de Estudos
Impacto em qualidade de software	7
Refatoração	5
Coocorrência	4
Total	16

Relação de Coocorrência

QP2(RSL). A literatura tem explorado coocorrências entre padrões de projeto e *bad smells*?

Relação	Quantidade de Estudos
Impacto em qualidade de software	7
Refatoração	5
Coocorrência	4
Total	16

Relação de Coocorrência

QP2.1(RSL). Quais *bad smells* são abordados pela literatura na identificação de coocorrências com padrões de projeto?

Bad Smells descritos por Brown et al. (1998)		
Anti Singleton	*Blob*	*Class Data Should Be Private*
Spaghetti Code	*Swiss Army Knife*	
Bad Smells descritos por Fowler & Beck (1999)		
Data Class	*Data Clumps*	*Duplicate Code*
Feature Envy	*Long Method*	*Long Parameter List*
Message Chains	*Refused Bequest*	*Speculative Generality*
Bad smells descritos por Lanza & Marinescu (2006)		
External Duplication	*God Class*	*Schizophrenic Class*

Relação de Coocorrência

QP2.2(RSL). Quais padrões de projeto são usados pela literatura na identificação de coocorrências com *bad smells*?

Jaafar et al. (2013)					
Command	*Composite*	*Decorator*	*Factory Method*	*Observer*	*Prototype*
Cardoso & Figueiredo (2014)					
Adapter	*Bridge*	*Command*	*Composite*	*Decorator*	*Factory Method*
Observer	*Prototype*	*Proxy*	*Singleton*	*State*	*Strategy*
Template Method	*Visitor*				
Jaafar et al. (2010)					
Command	*Composite*	*Decorator*	*Factory Method*	*Observer*	*Prototype*
Walter & Alkhaeir (2016)					
Adapter	*Bridge*	*Command*	*Composite*	*Decorator*	*Factory Method*
Observer	*Prototype*	*Proxy*	*Singleton*	*State*	*Strategy*
Template Method	*Visitor*				

Considerações

- Em geral, os trabalhos foram classificados em três categorias
  - Impacto em qualidade de software
  - Refatoração
  - Coocorrência

- Relações de coocorrência são poucos abordadas
  - São recentes, a partir de 2013

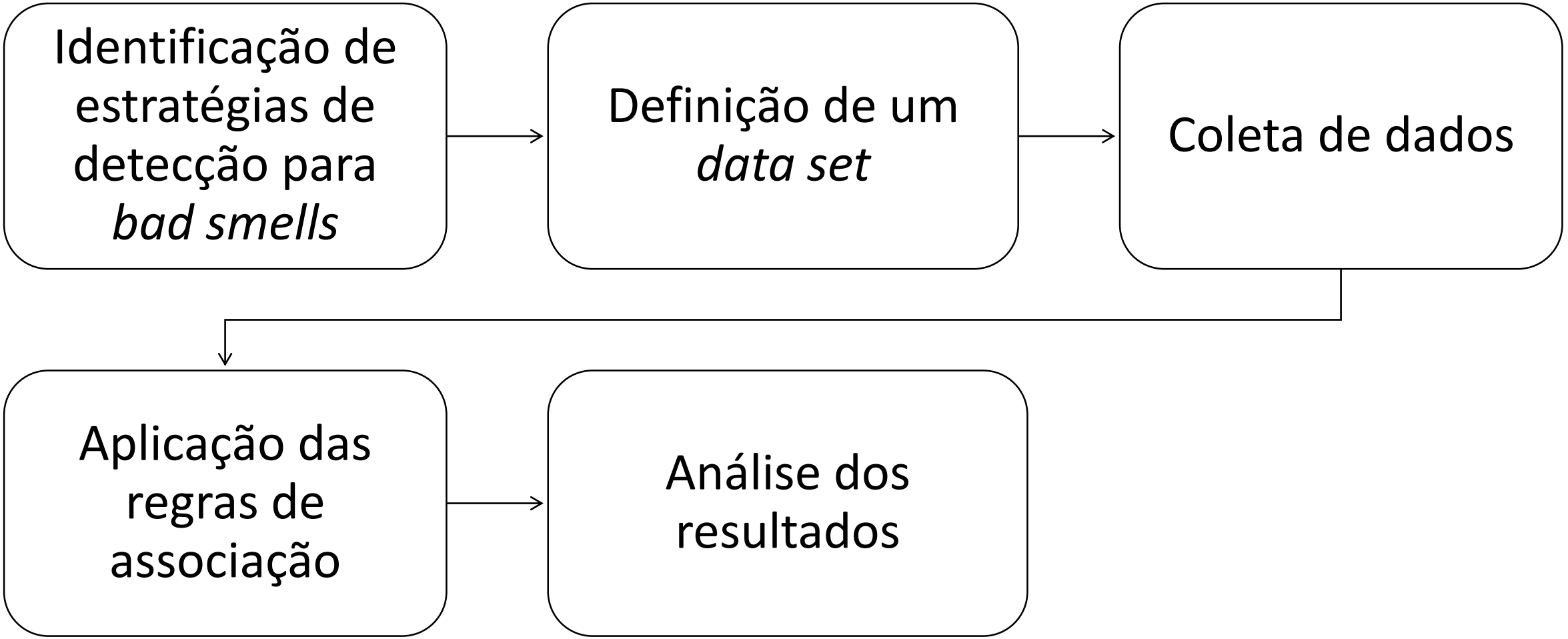
Considerações

- *Bad smells* abordados em estudos de coocorrência
  - Conjunto descrito por Fowler & Beck (1999) são os mais abordados
  - Existência de ferramentas e estratégias de detecção

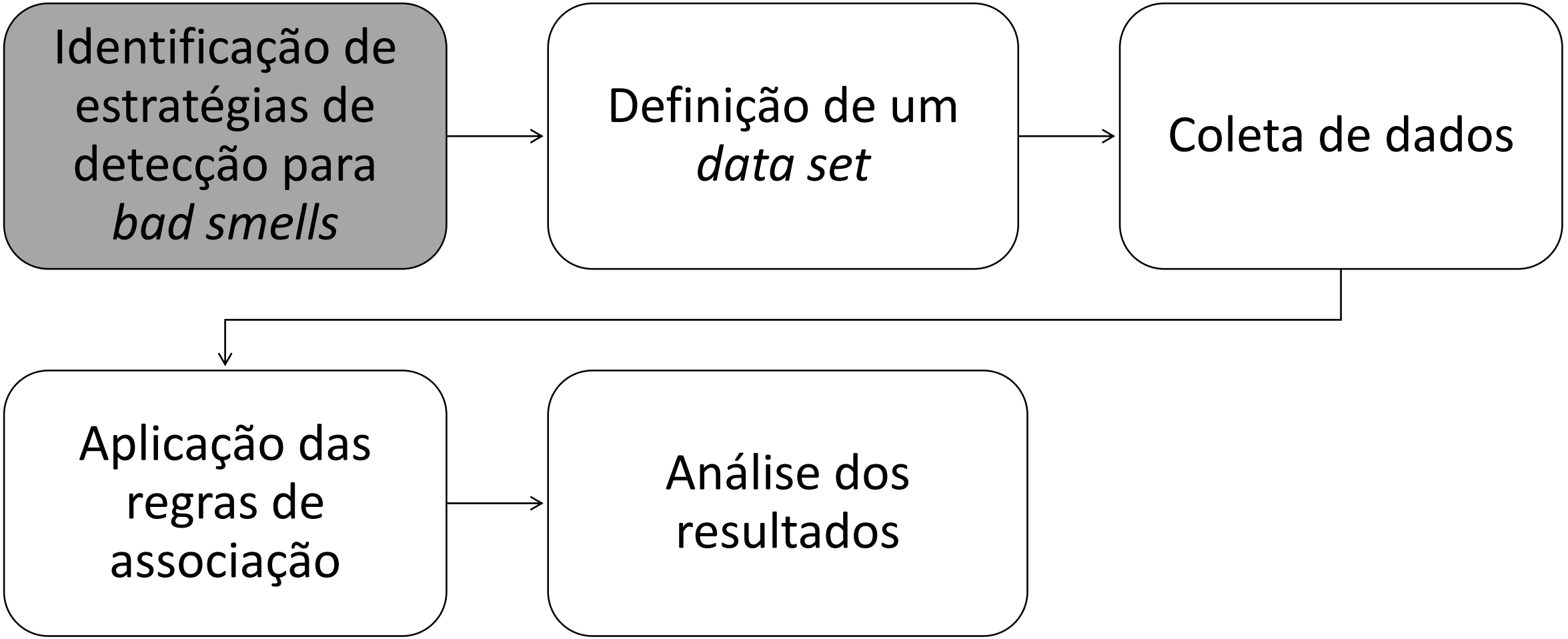
- Padrões de projeto abordados em estudos de coocorrência
  - Conjunto de padrões de projeto *GOF*
  - São os mais conhecidos e utilizados
  - Existência de ferramentas para detecção de instâncias.

Metodologia

Etapas da Metodologia



Etapas da Metodologia

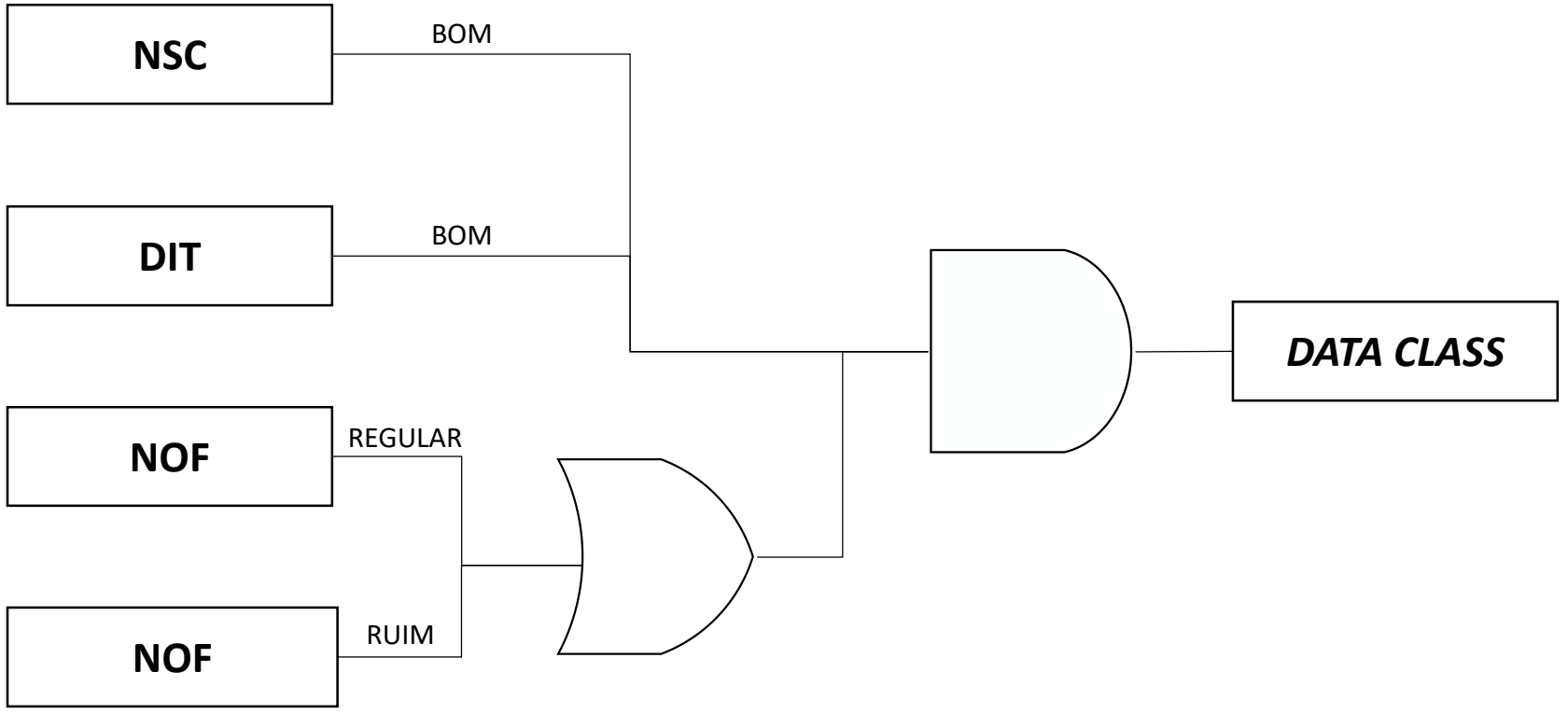


Identificação de Estratégias de Detecção para os *Bad Smells*

- Considerou-se as estratégias proposta por Souza (2016)
  - Aplicam métricas de software bem conhecidas
  - Foram previamente avaliadas
  - Resultados indicaram uma boa precisão para detecção de *bad smells*

- Valores referência das estratégias de detecção
  - Baseado no catálogo proposto por Filó et al. (2015)
  - Uso de três faixas: **BOM**, **REGULAR** e **RUIM**

Identificação de Estratégias de Detecção para os *Bad Smells*



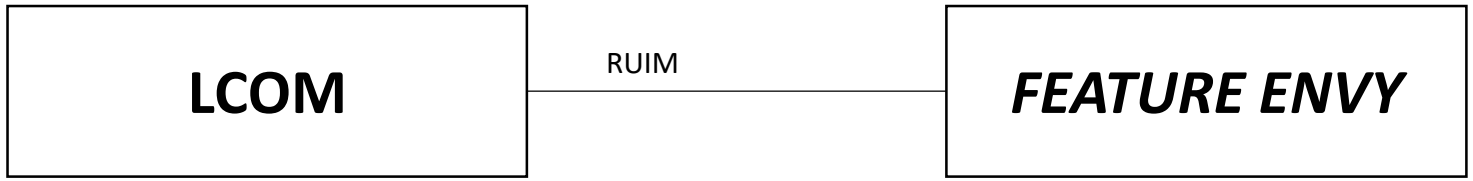
Métricas

Número de Filhas (**NSC**)

Profundidade da Árvore de Herança (**DIT**)

Número de Atributos (**NOF**)

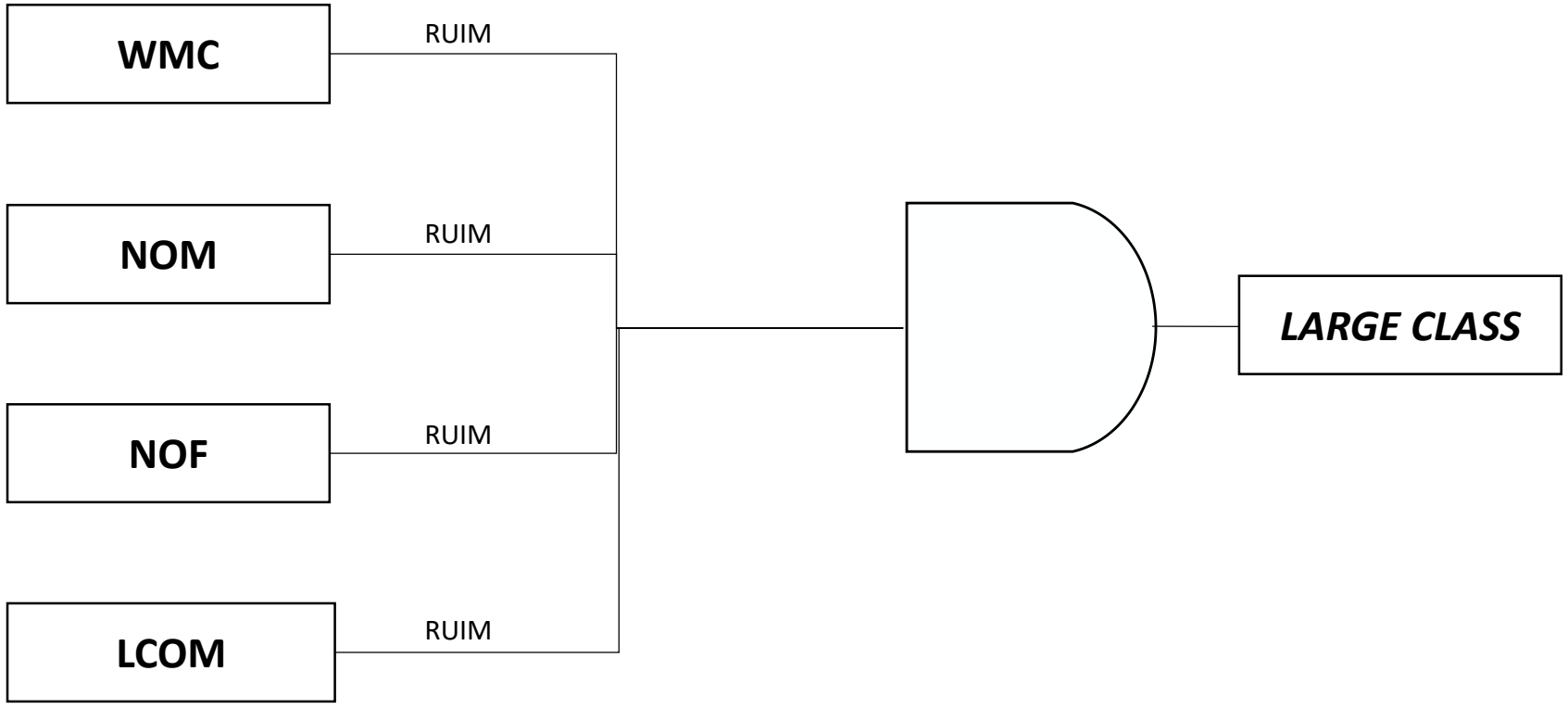
Identificação de Estratégias de Detecção para os *Bad Smells*



Métrica

Falta de Coesão dos Métodos (**LCOM**)

Identificação de Estratégias de Detecção para os *Bad Smells*



Métricas

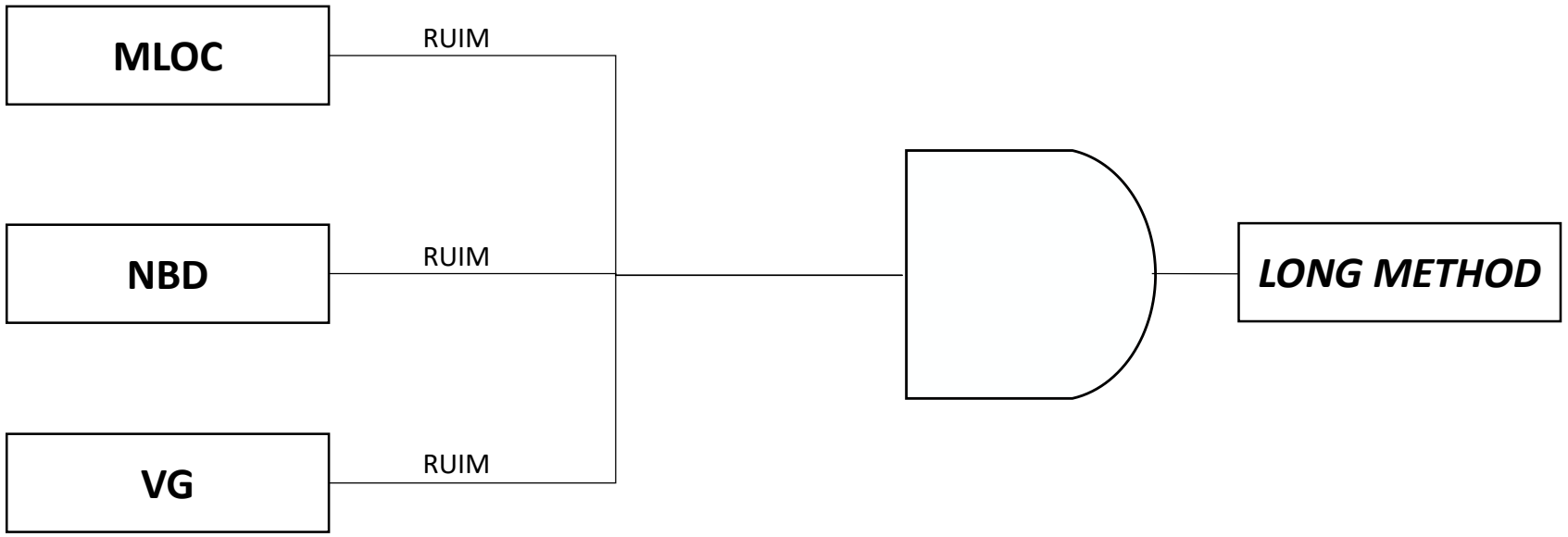
Métodos Ponderados por Classe (**WMC**)

Número de Métodos (**NOM**)

Número de Atributos (**NOF**)

Falta de Coesão dos Métodos (**LCOM**)

Identificação de Estratégias de Detecção para os *Bad Smells*



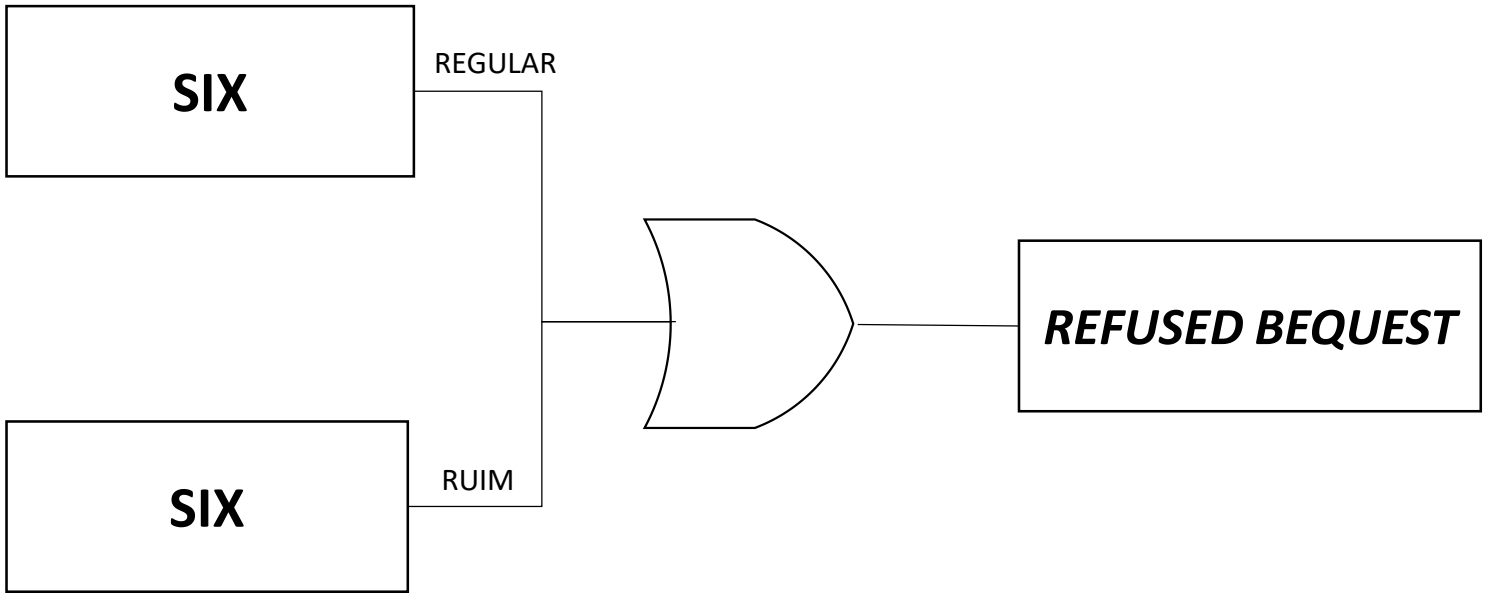
Métricas

Linhas de Código por Método (**MLOC**)

Profundidade de Blocos Aninhados (**NBD**)

Complexidade Ciclomática de McCabe (**VG**)

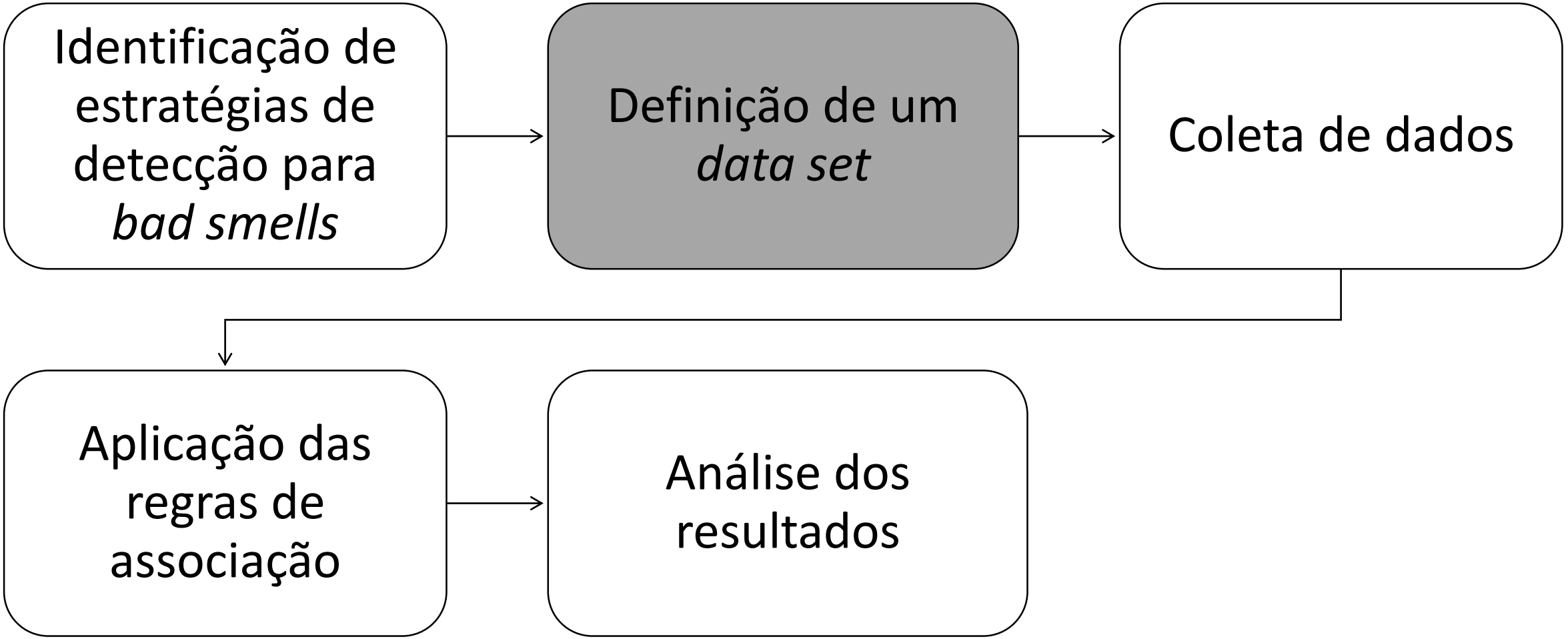
Identificação de Estratégias de Detecção para os *Bad Smells*



Métrica

Índice de Especialização (**SIX**)

Etapas da Metodologia



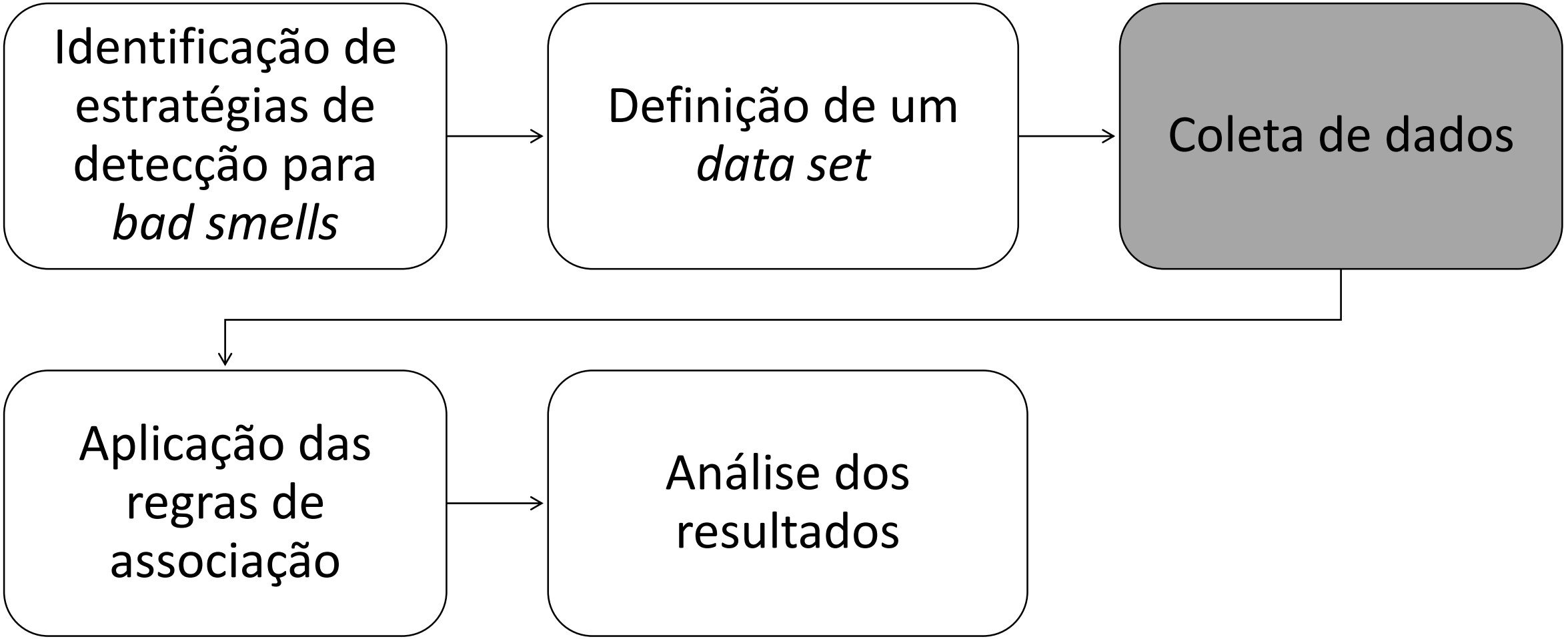
Definição do *Data Set*

- *Qualitas.class Corpus* (Terra et. al., 2013)

- Sistemas usados no *data set*
  - Hibernate 4.2.0
  - JHotDraw 7.5.1
  - Kolmafia 17.3
  - Webmail 0.7.10
  - Weka 3.6.9

- Kolmafia não pertence ao *Qualitas.class Corpus*

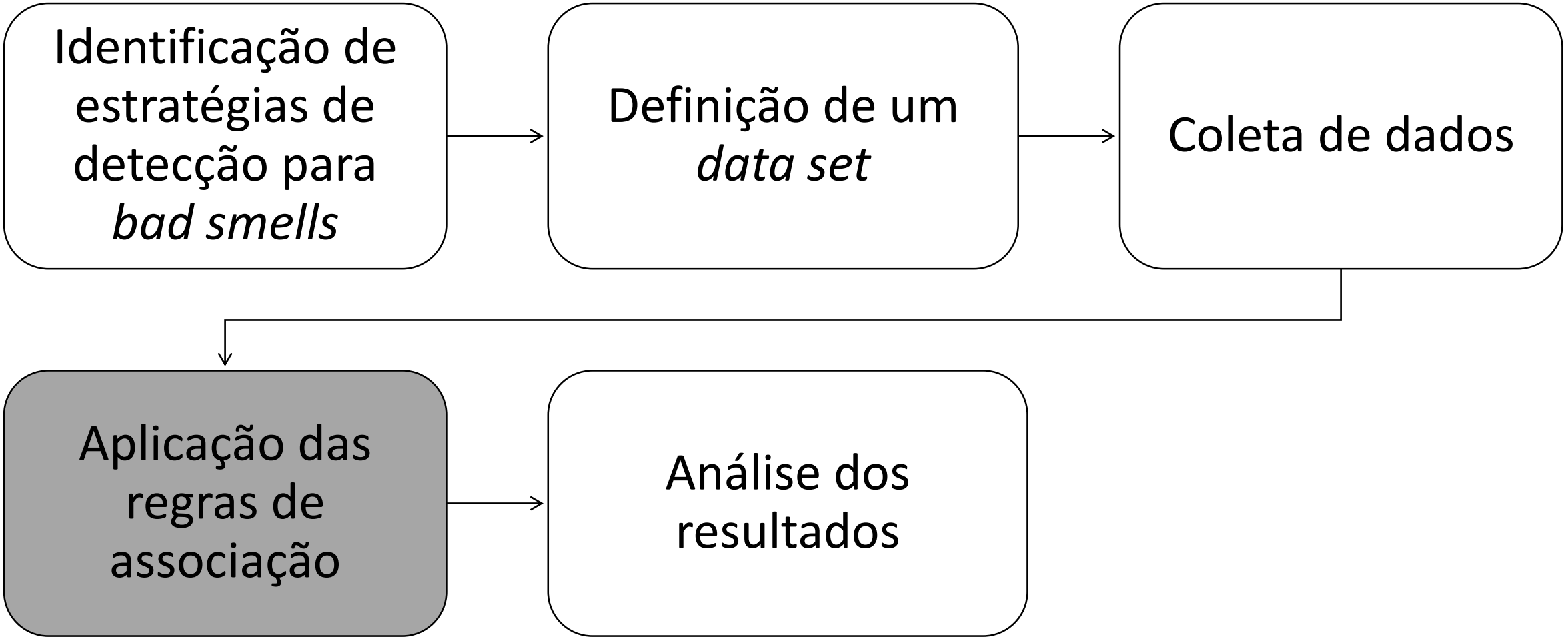
Etapas da Metodologia



Coleta de Dados

- Métricas
  - *Qualitas.class Corpus*
  - *Eclipse* 4.2 *Juno*
  - *Metrics* 1.3.6
- Padrão de Projeto
  - *Design Pattern Detection using Similarity Scoring* (Tsantalis et al. 2006)
- Bad Smells
  - *RAFTool* (Filó et al. 2014)
- Coocorrências
  - *Design Pattern Smell*

Etapas da Metodologia



Regras de Associação

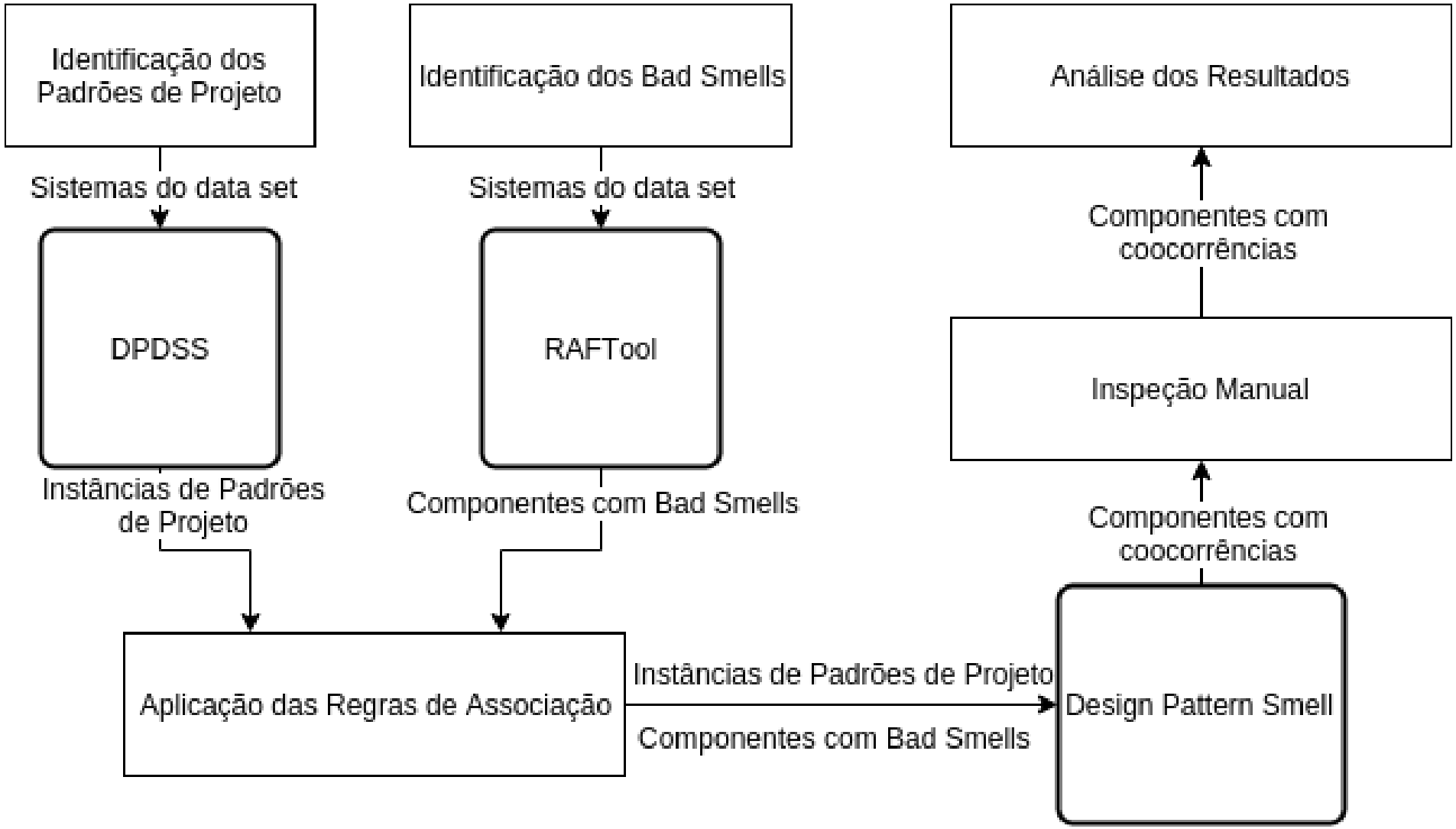
- Combinação de itens e extração de conhecimento
- Estudos anteriores tem usado esse método
- Regras
  - Suporte
  - Confiança
  - Convicção

Regras de Associação

- Transação
  - Classe existente no sistema analisado
- Antecedente
  - Padrão de projeto
- Consequente
  - *Bad smell*
- **Antecedente → Consequente**

Etapas da Metodologia

Método de Análise dos Resultados



Design Pattern Smell

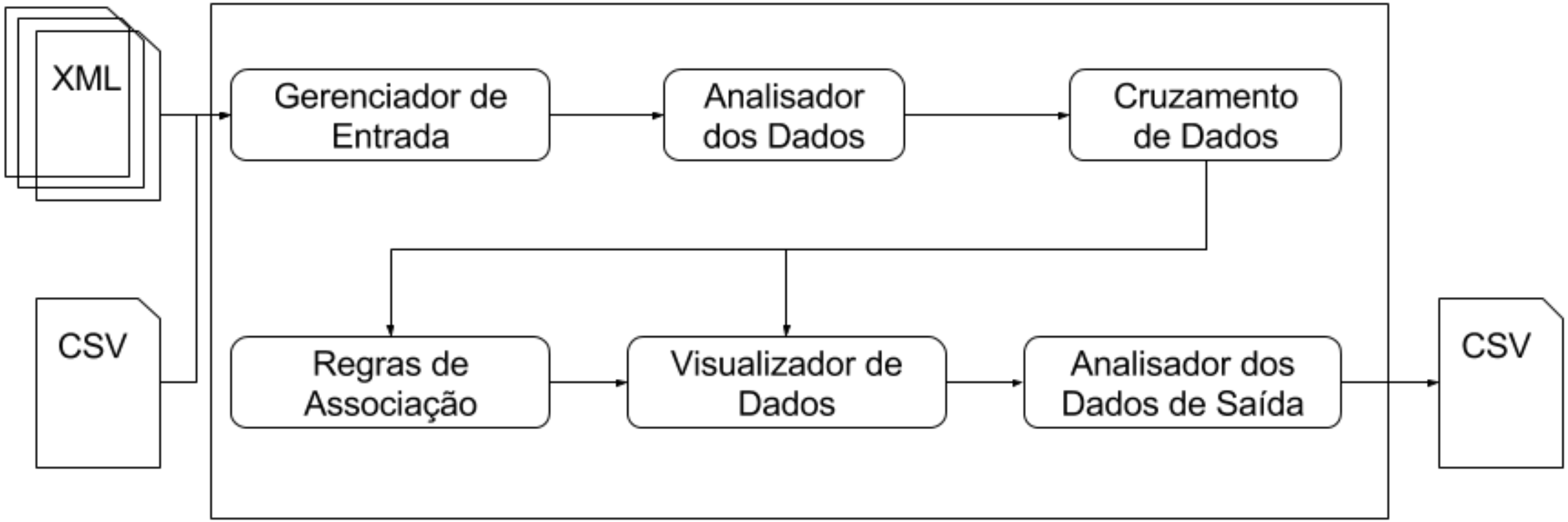
Design Pattern Smell

- Estudos anteriores têm investigado relações de coocorrência
  - Não propuseram ferramentas para detecção dessas relações
- *Design Pattern Smell* suporta detecção de coocorrências
- Baseada em informações extraídas do código fonte de um sistema
- Suporta a aplicação de regras de associação nos dados fornecidos

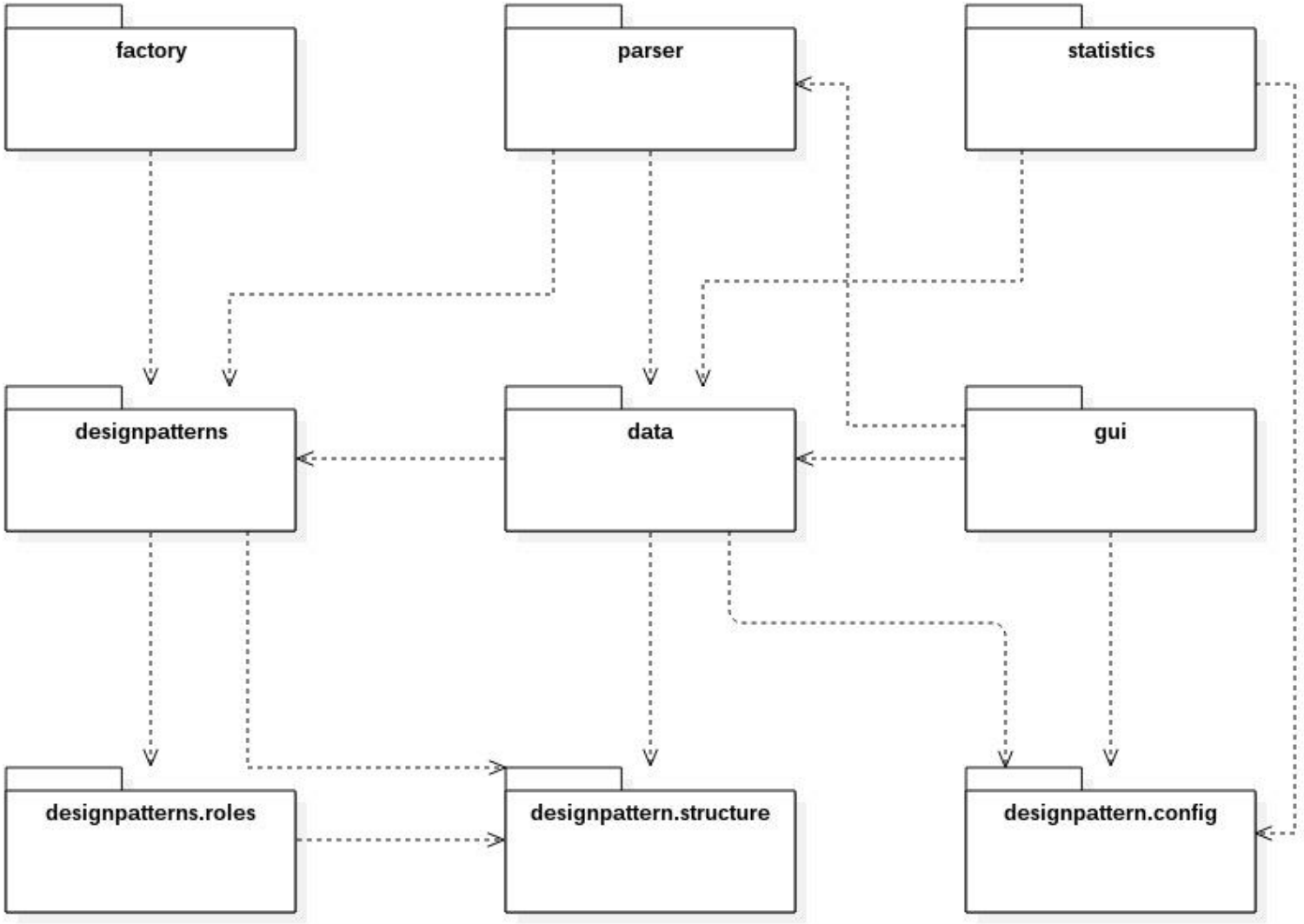
Principais Funcionalidades

- Importação das Instâncias de Padrões de Projeto Computadas
- Importação dos Artefatos com *Bad Smells*
- Aplicação das Regras de Associação
- Geração, Visualização e Exportação
- Gerenciamento de Dados
- Ajuda

Arquitetura



Organização Interna



Implementação

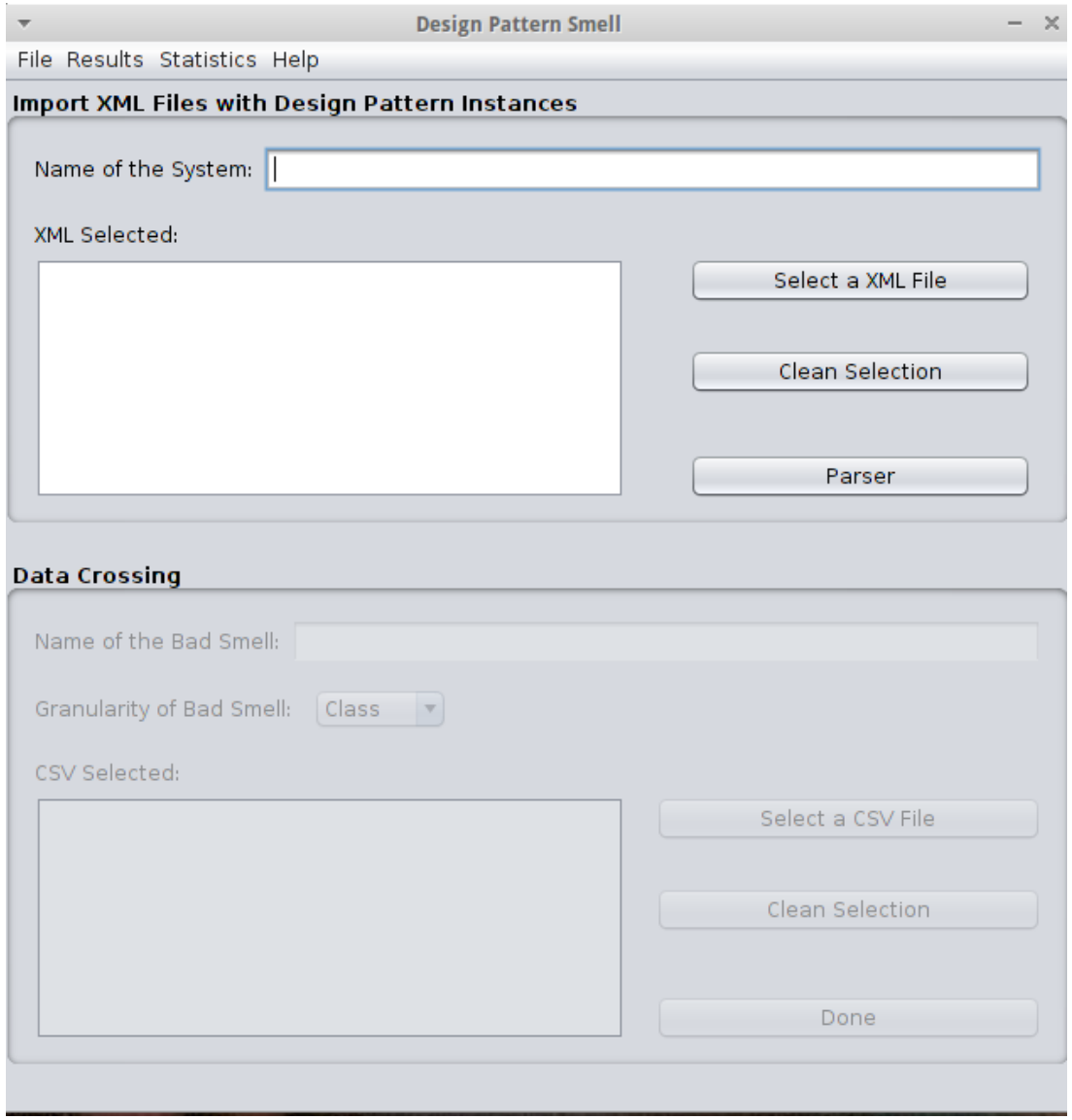
- Linguagem de Programação Java
  - *JDK* 1.7
  - *API Java Swing*

- *XML* parser
  - *API JDOM*

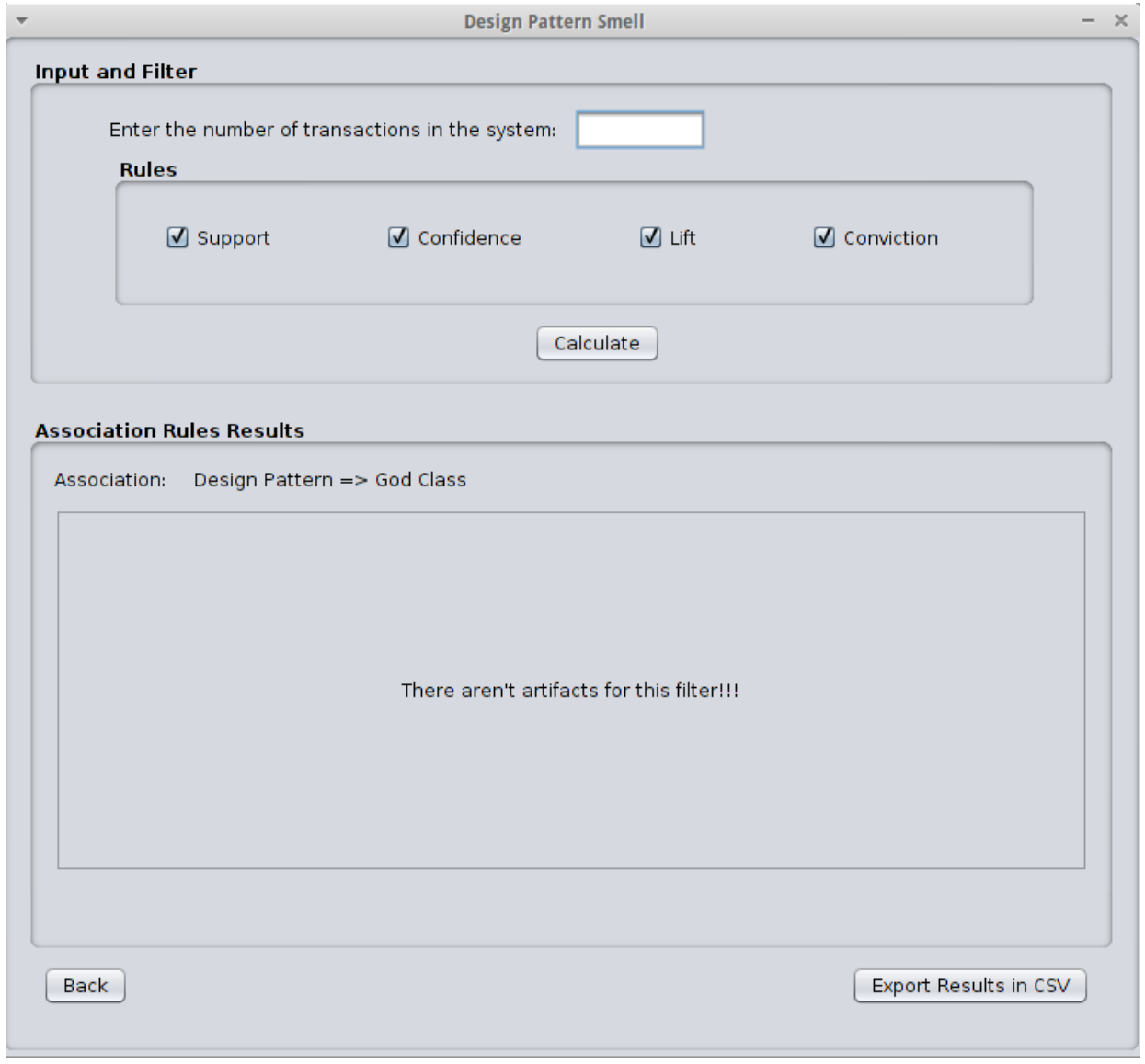
- Descrição das Regras de Associação
  - *API JLaTeXMath* 1.0.3

- *IDE NetBeans* 8.0.2

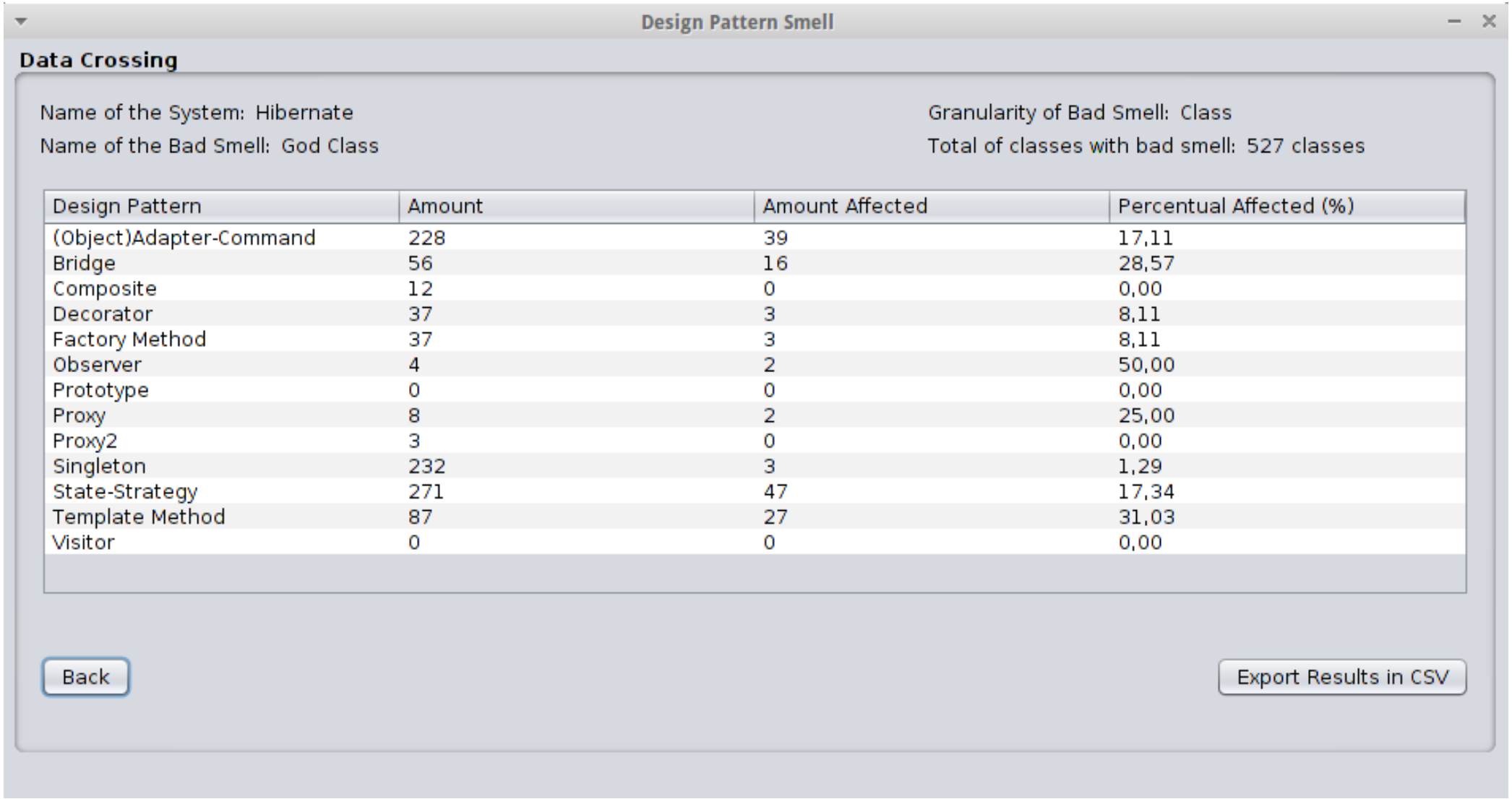
Tela Principal



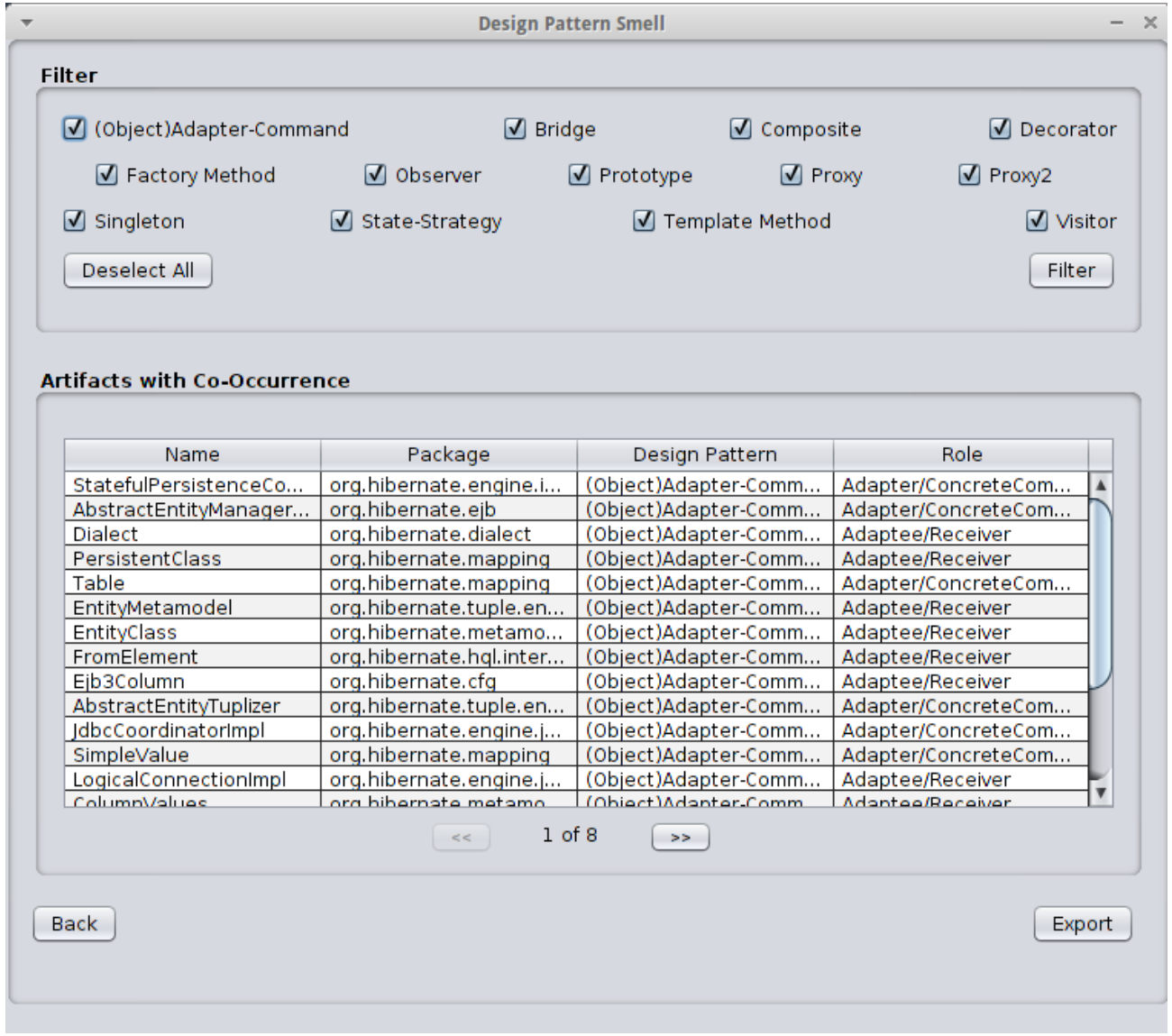
Tela de Aplicação das Regras de Associação



Tela de Resultados



Tela de Visualização das Coocorrências



Considerações

- Design Pattern Smell
  - Disponível na versão 1.0
  - Suporta 14 padrões de projeto GOF
- Requisito para uso da Design Pattern Smell
  - Máquina Virtual Java 1.7 ou superior instalada
- Desdobramentos futuros
  - Suporte aos 23 padrões de projeto GOF
  - Adesão de um módulo de análise estática

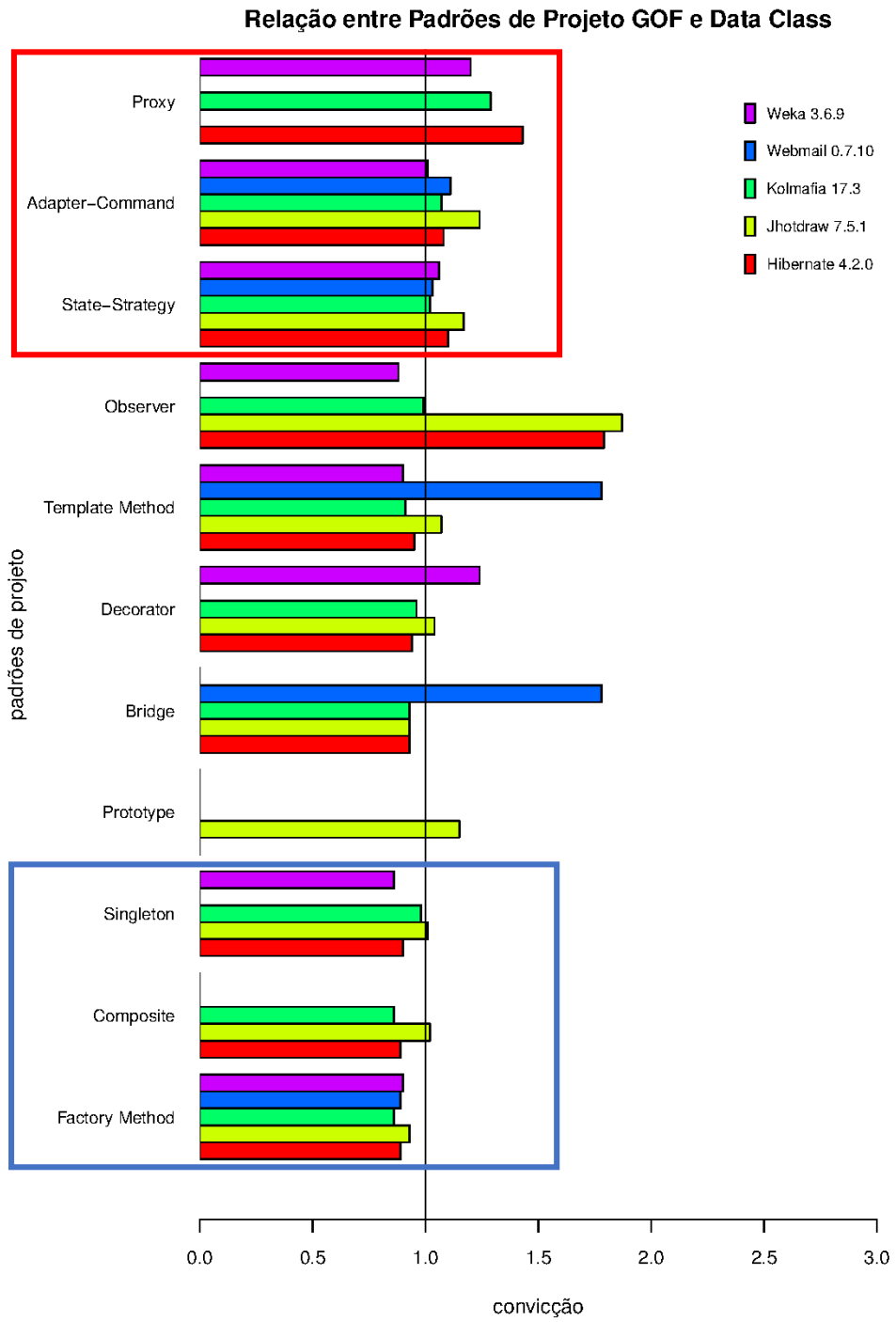
Estudo de Caso

Descrição do Estudo de Caso

- Coleta das instâncias de padrões de projeto nos sistemas do *data set*
- Coleta dos *bad smells* nos sistemas do *data set*
- Identificação das coocorrências
- Aplicação das regras de associação para identificação da intensidade das relações

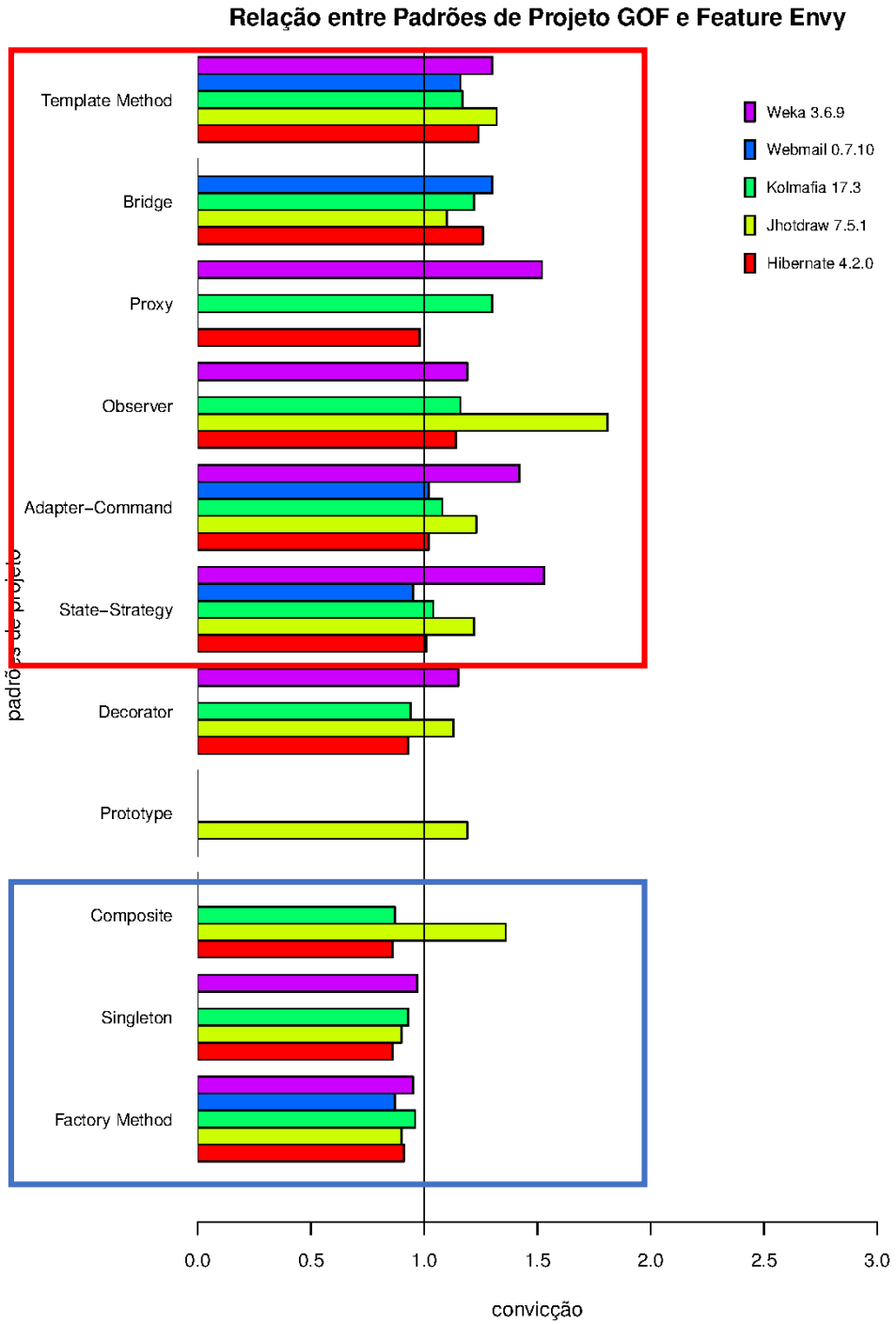
Resultados

Data Class



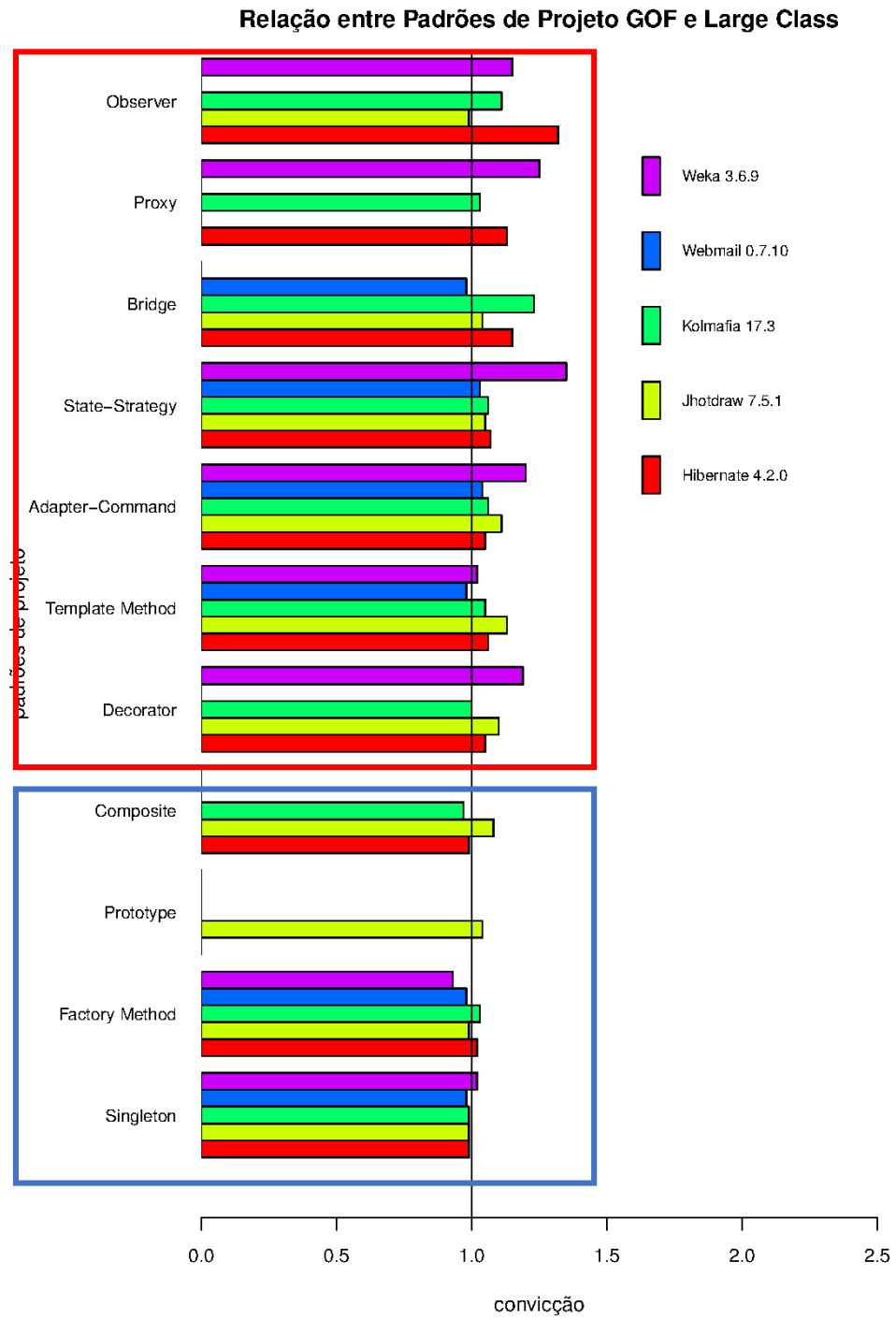
Resultados

Feature Envy



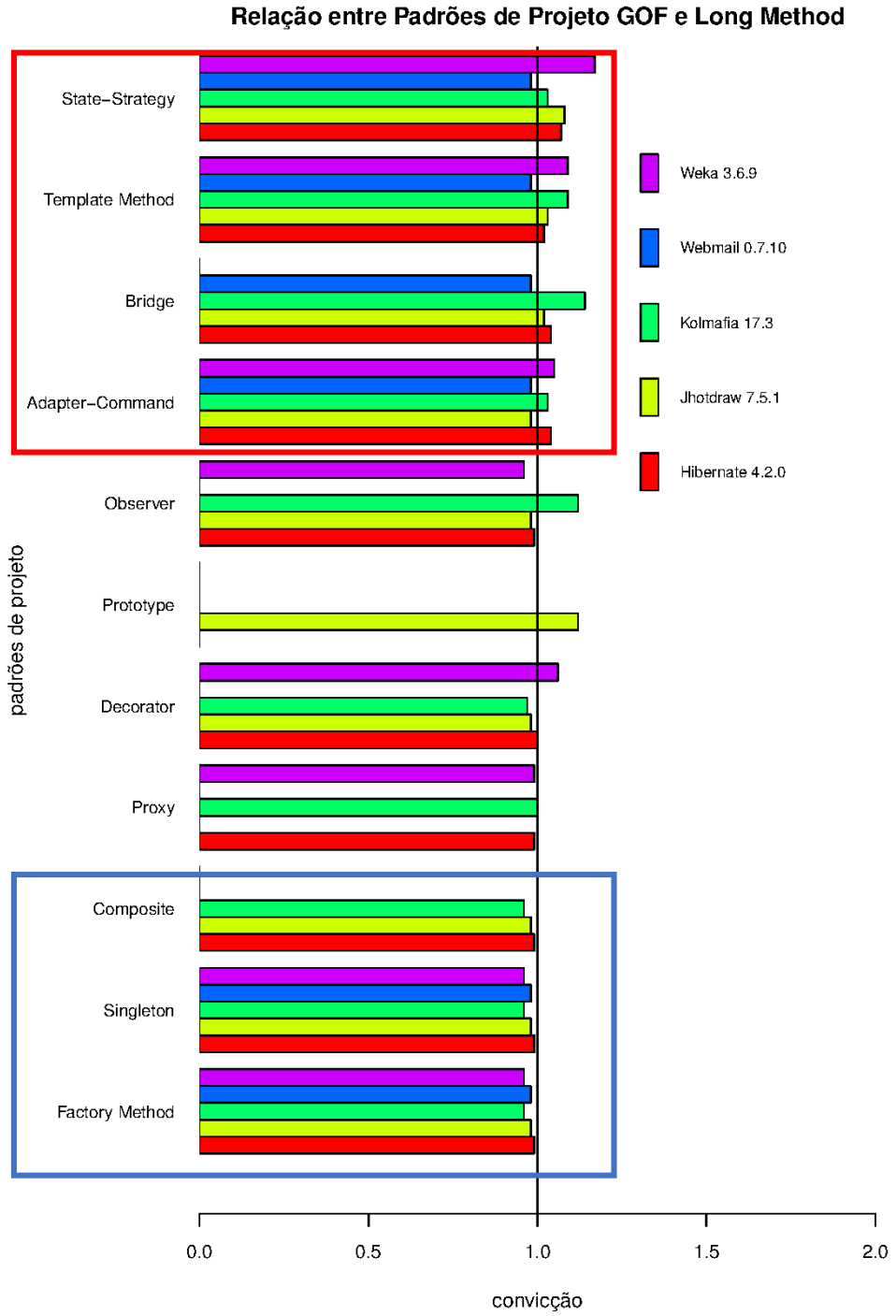
Resultados

Large Class



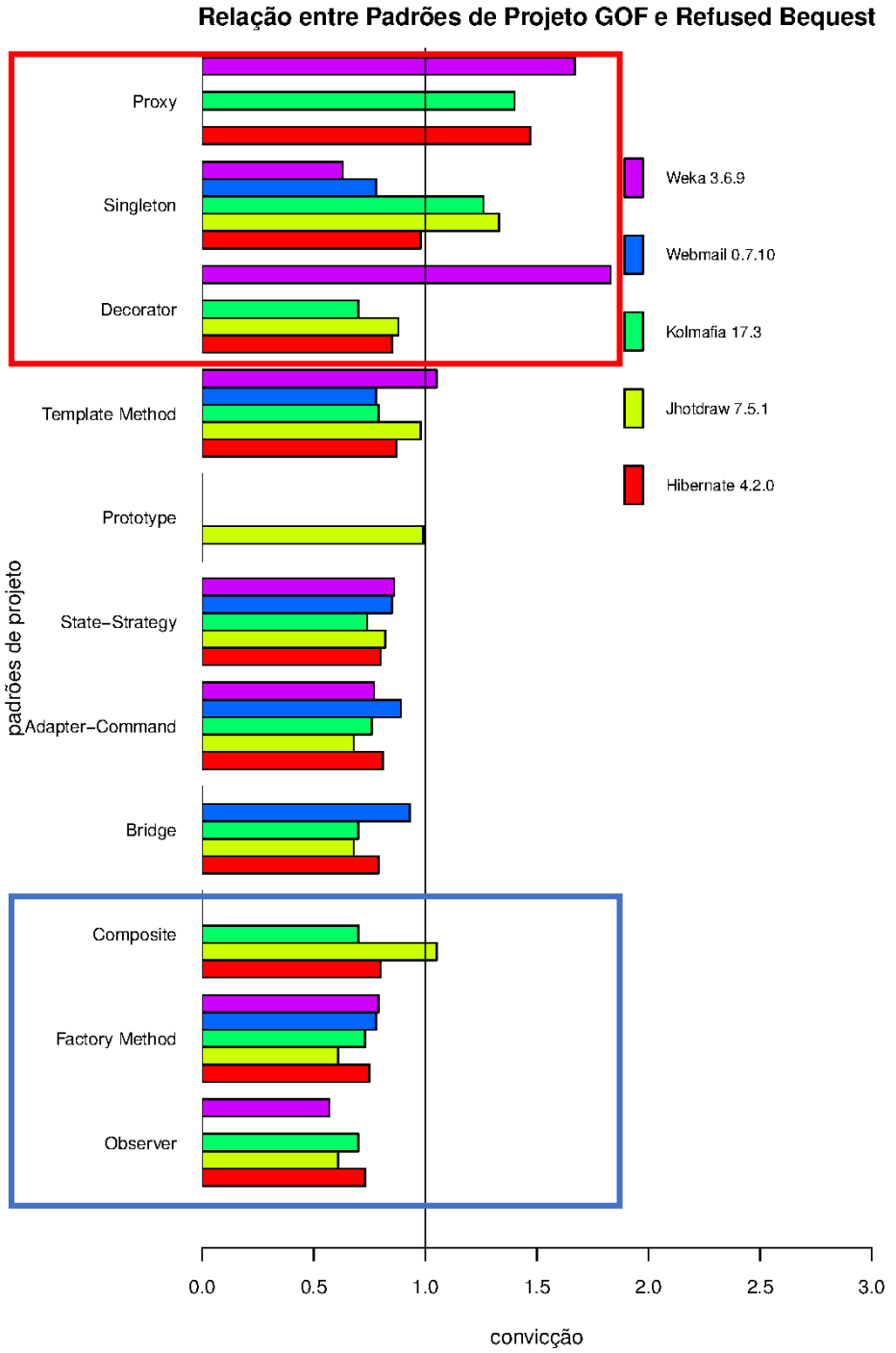
Resultados

Long Method



Resultados

Refused Bequest



Questão de Pesquisa 1

QP1. Os padrões de projeto definidos no catálogo *GOF* evitam a ocorrência de *bad smells* em software?

Questão de Pesquisa 1

- Alguns padrões de projeto apresentaram baixa coocorrência
  - *Composite*
  - *Factory Method*

- *Composite* e *Factory Method* são padrões intrinsicamente modulares

- *Singleton* apresentou baixa coocorrência para a maioria dos padrões
  - Inconclusivo em relação ao *Long Method*

- Padrões de projeto não necessariamente evitam ocorrência de *bad smells*

Questão de Pesquisa 2

QP2. Quais padrões de projeto do catálogo *GOF* apresentaram coocorrência com *bad smells*?

Questão de Pesquisa 2

Bad Smell	Padrão de Projeto		
Data Class	Proxy	Adapter-Command	State-Strategy
Feature Envy	Template Method Observer	Bridge Adapter-Command	Proxy State-Strategy
Large Class	Observer Adapter-Command Decorator	Proxy Bridge	State-Strategy Template Method
Long Method	State-Strategy Bridge	Template Method	Adapter-Command
Refused Bequest	Proxy	Singleton	

Questão de Pesquisa 3

QP3. Quais são as situações mais comuns em que *bad smells* aparecem em sistemas de software que aplicam os padrões de projeto *GOF*?

Proxy → Data Class

- Padrão *proxy* tem como propósito controlar acesso a objetos

- Classes *RealSubject* apresentaram essa coocorrência

- Situações contribuíram para essa coocorrência
  - Implementação da classe *RealSubject* dentro do padrão *Proxy*
  - Alta quantidade de dados e pouca funcionalidade sobre eles

Template Method → Feature Envy

- *Template Method* define um esqueleto de algoritmo por meio de um método e permite que subclasses redefinam características de objetos

- Classes *AbstractClass* apresentaram essa coocorrência

- Situação que contribuiu para essa coocorrência
  - Métodos de mesmo interesse da classe *AbstractClass* implementados em outras classes

Observer → Large Class

- *Observer* define uma dependência do tipo um para muitos entre objetos
- Classes *Subject* apresentaram essa coocorrência
- Situações que contribuiu para essa coocorrência
  - Modelagem inadequada da classe *Subject*
  - Uso de uma única classe *Subject* para observadores de interesses diferentes

Strategy → Long Method

- *Strategy* define uma família de algoritmos
- Classes *Context* apresentaram essa coocorrência
- Situação que contribuiu para essa coocorrência
  - Excesso de código utilizado na implementação das estratégias
  - Uso de estruturas condicionais para definição das estratégias ao invés de polimorfismos

Proxy → Refused Bequest

- Padrão *proxy* tem como propósito controlar acesso a objetos
- Classes *Proxy* apresentaram essa coocorrência
- Situações contribuíram para essa coocorrência
  - Uso de herança como mecanismo de reuso
  - Posicionamento de componentes em um nível inadequado na árvore de herança

Considerações

- Padrões de projeto não necessariamente evitam ocorrência de *bad smells*
  - *Composite*, *Factory Method* e *Singleton* apresentaram baixa coocorrência
  - Resultado para o padrão *Singleton* contradiz estudos anteriores

- Diversas coocorrências foram extraídas

- Mal uso e implementação inadequada contribuíram para o surgimento dessas relações

Ameaças à Validade

Ameaças à Validade

- *String* de Busca criada para a revisão sistemática
- Bases Eletrônicas não garantem que todos os estudos relevantes sejam retornados
- Linguagem dos documentos restrito ao inglês
- Leitura reduzida dos estudos

Ameaças à Validade

- Resultados do Estudo de Caso não podem ser generalizados
- Uso de ferramentas na coleta dos dados
- Inspeção manual

Conclusão

Conclusão

- Constatou-se que estudos sobre coocorrência entre padrão e *bad smell* é pouco explorado na literatura

- Padrões de projeto não evitam ocorrência de *bad smells*

- Mal uso e implementação inadequada impactam no surgimento das coocorrências

Conclusão

- Contribuições
  - Revisão Sistemática da Literatura sobre padrões de projeto e *bad smells*
  - Ferramenta *Design Pattern Smell* para detecção de coocorrências
  - Avaliação de coocorrência entre padrão de projeto e *bad smell* por meio de um Estudo de Caso

- Publicação
  - Sousa, B. L.; Bigonha, M. A. S.; Ferreira, K. A. M. *Evaluating Co-Occurrence of GOF Design Patterns with God Class and Long Method Bad Smells*. In proceedings of the Brazilian Symposium on Information Systems, Lavras, Brazil, pages 396–403, 2017.

Conclusão

- Trabalhos Futuros
  - Estender essa pesquisa a uma quantidade maior de sistemas de software
  - Investigar coocorrências de padrões de projeto com outros tipos de *bad smells*
  - Investigar a relação de padrões de projeto, *bad smell* e falhas de software

PPGCC
PROGRAMA DE PÓS-GRADUAÇÃO
EM CIÊNCIA DA COMPUTAÇÃO
DCC-UFMG

OBRIGADO!

Bruno Luan de Sousa

Orientadora: Mariza Andrade da Silva Bigonha (UFMG)
Coorientadora: Kécia Aline Marques Ferreira (CEFET-MG)

07 de Julho de 2017

LLP Laboratório de Linguagens de Programação