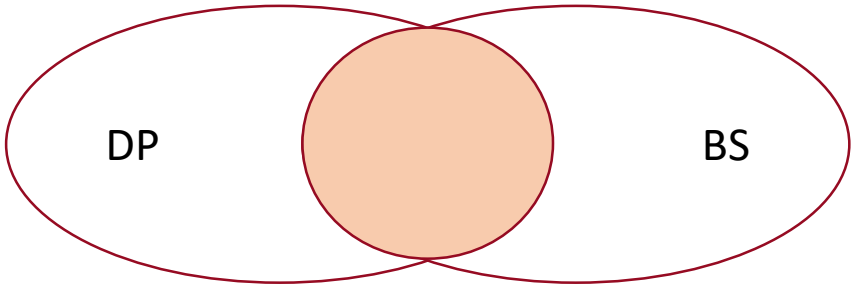
# Evaluating Co-Occurrence of GOF Design Patterns with God Class and Long Method Bad Smells

Bruno L. Sousa (DCC - UFMG)
Mariza A. S. Bigonha (DCC - UFMG)
Kecia A. M. Ferreira (DECOM – CEFET-MG)

# Introduction

- Design pattern is a general solution to a recurring problems
- Bad smells are symptoms of problems in the source code or design of software systems
- **Problem:** Relationship between design patterns and bad smells
- Co-Occurrences

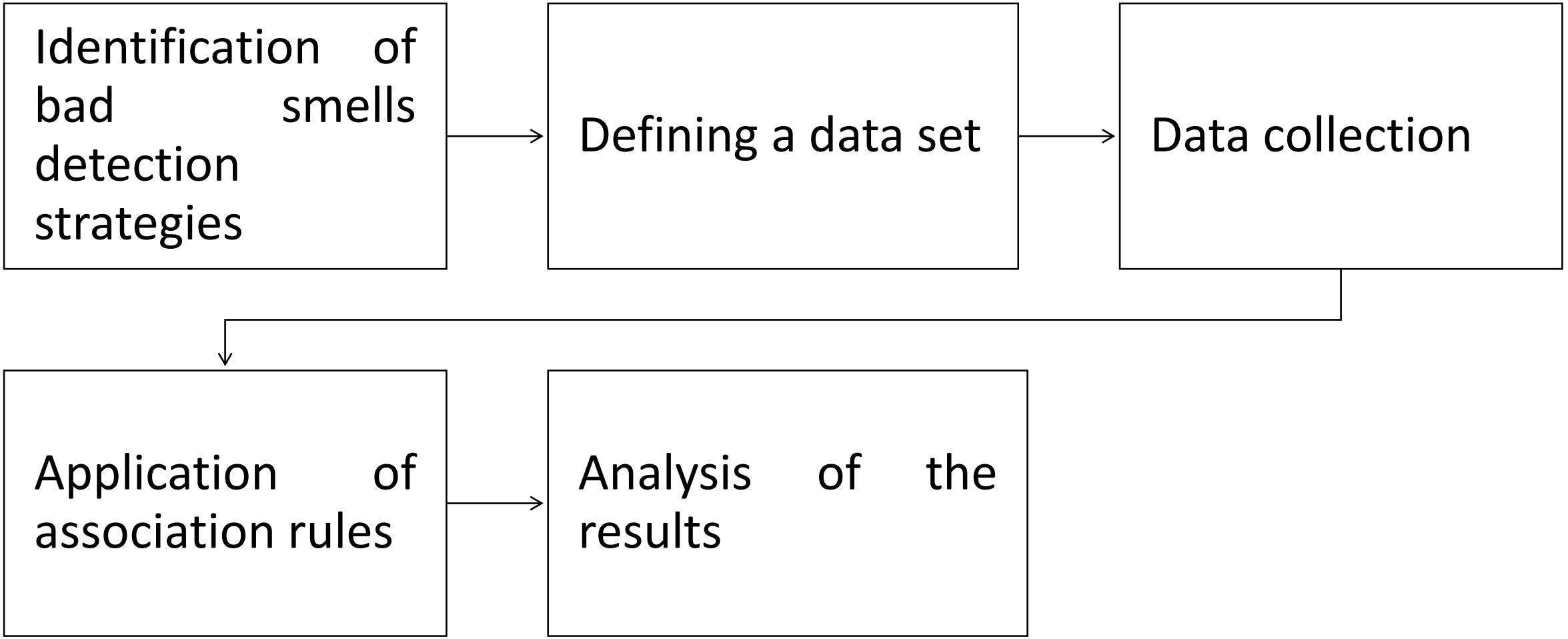


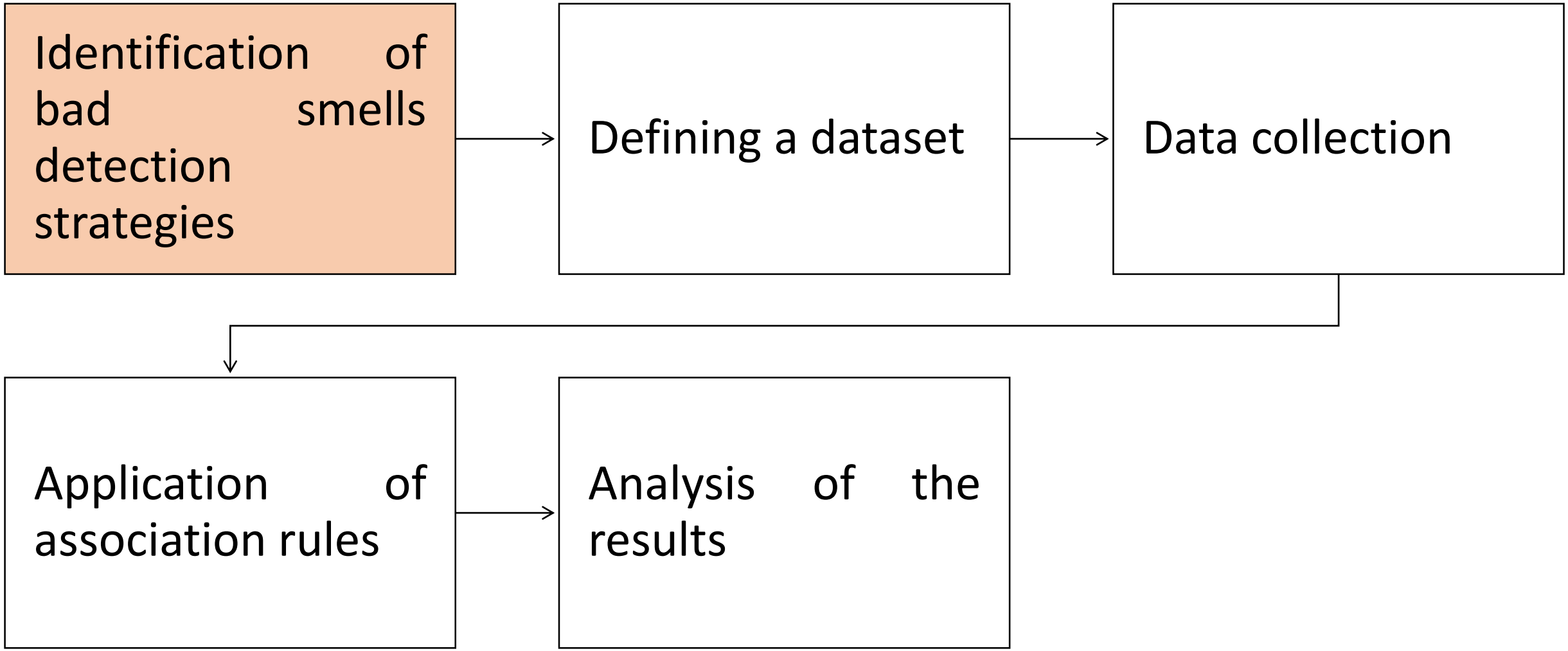
# Goal

- General Goal
  - Study the occurrences of God Class and Long Method bad smells in software systems developed with design patterns

- Research Questions
  - **RQ1.** Do the design patterns defined in GOF catalog avoid the occurrence of God Class and Long Method bad smells in software?
  - **RQ2.** Which design patterns of GOF catalog present co-occurrence with God Class and Long Method bad smells?
  - **RQ3.** What are the more common situations in which the God Class and Long Method bad smells appear in software systems that apply GOF design patterns?

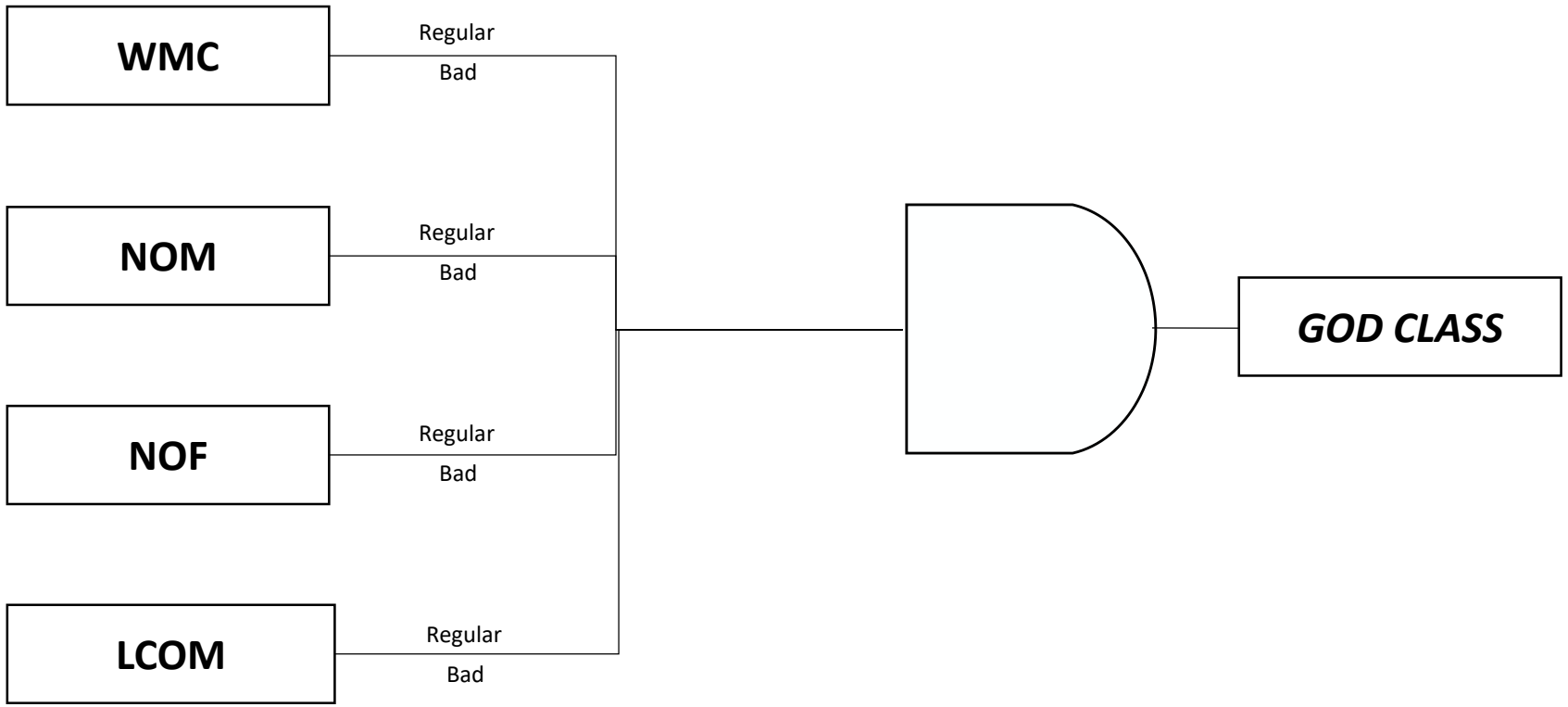
# Methodology




Identification of bad smells detection strategies
Defining a dataset
Data collection
Application of association rules
Analysis of the results

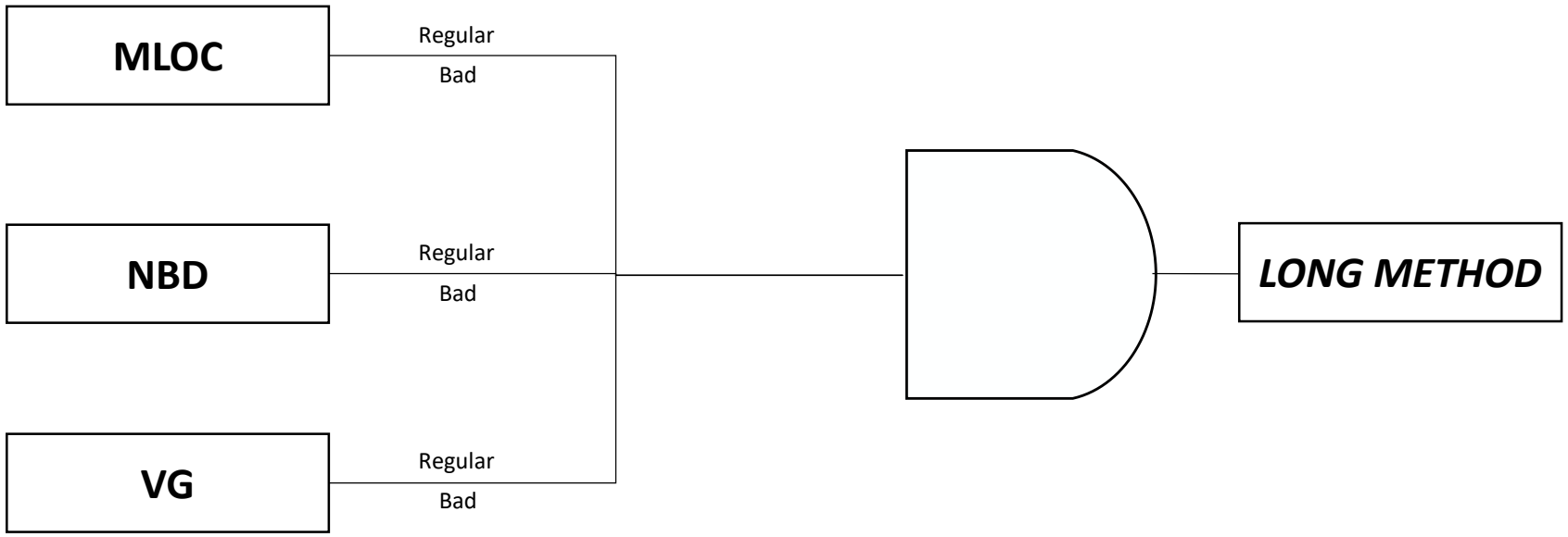
# Identification of Bad Smells Detection Strategies

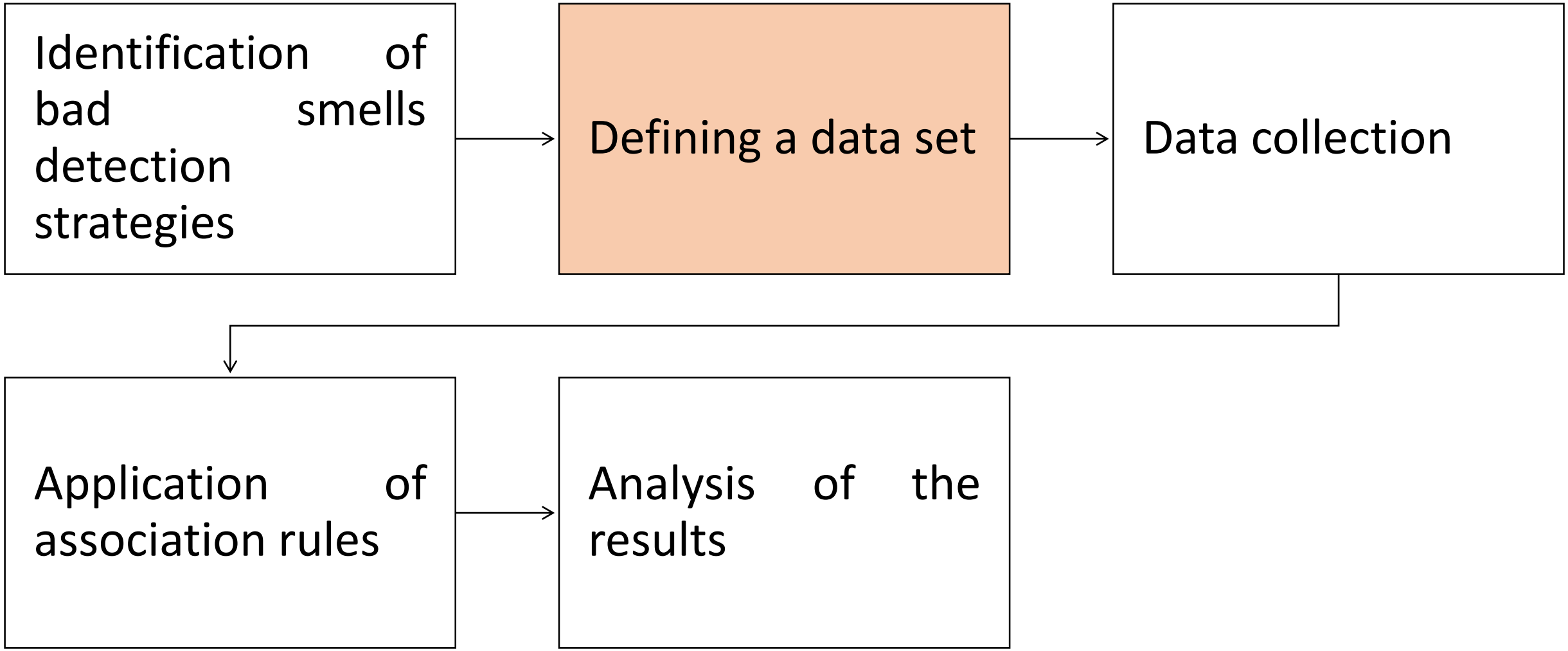
God Class



# Identification of Bad Smells Detection Strategies

Long Method



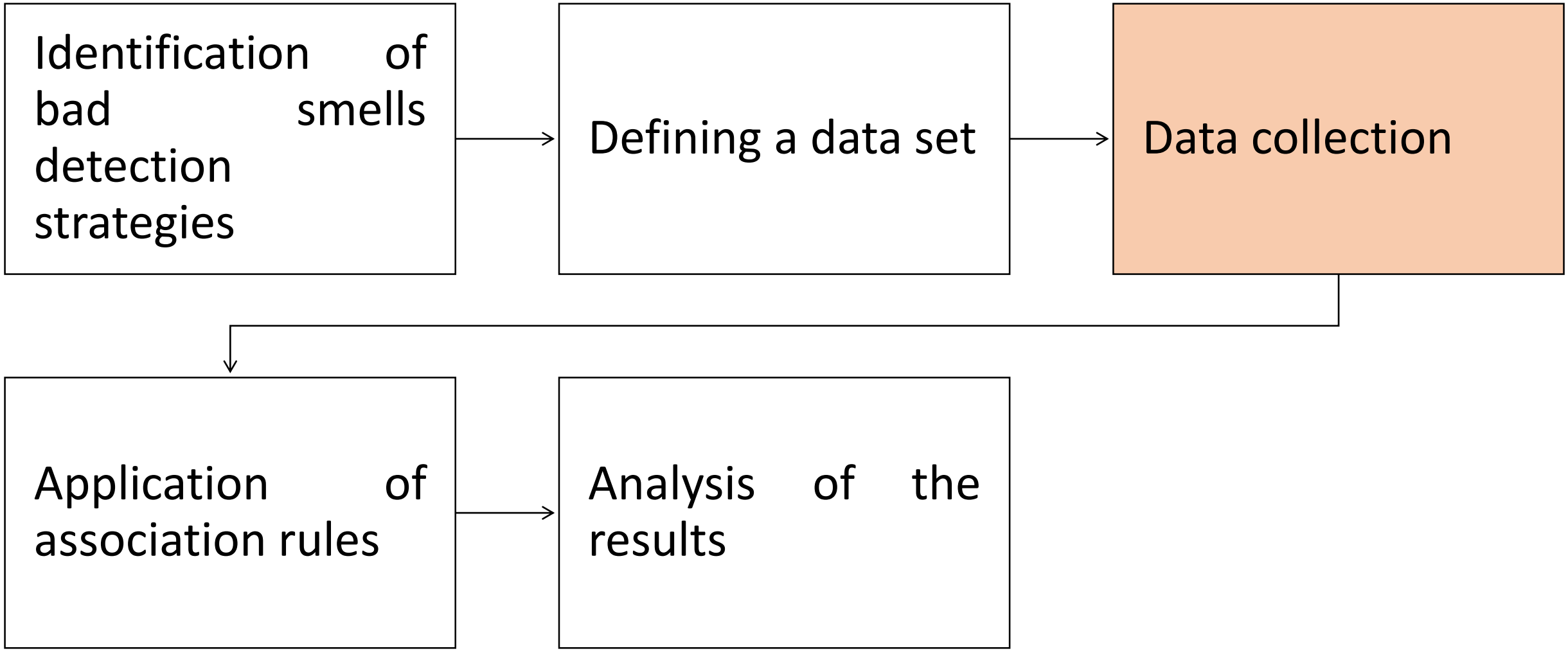

Identification of bad smells detection strategies
Defining a data set
Data collection
Application of association rules
Analysis of the results

# Defining a Data Set

- Qualitas.class Corpus

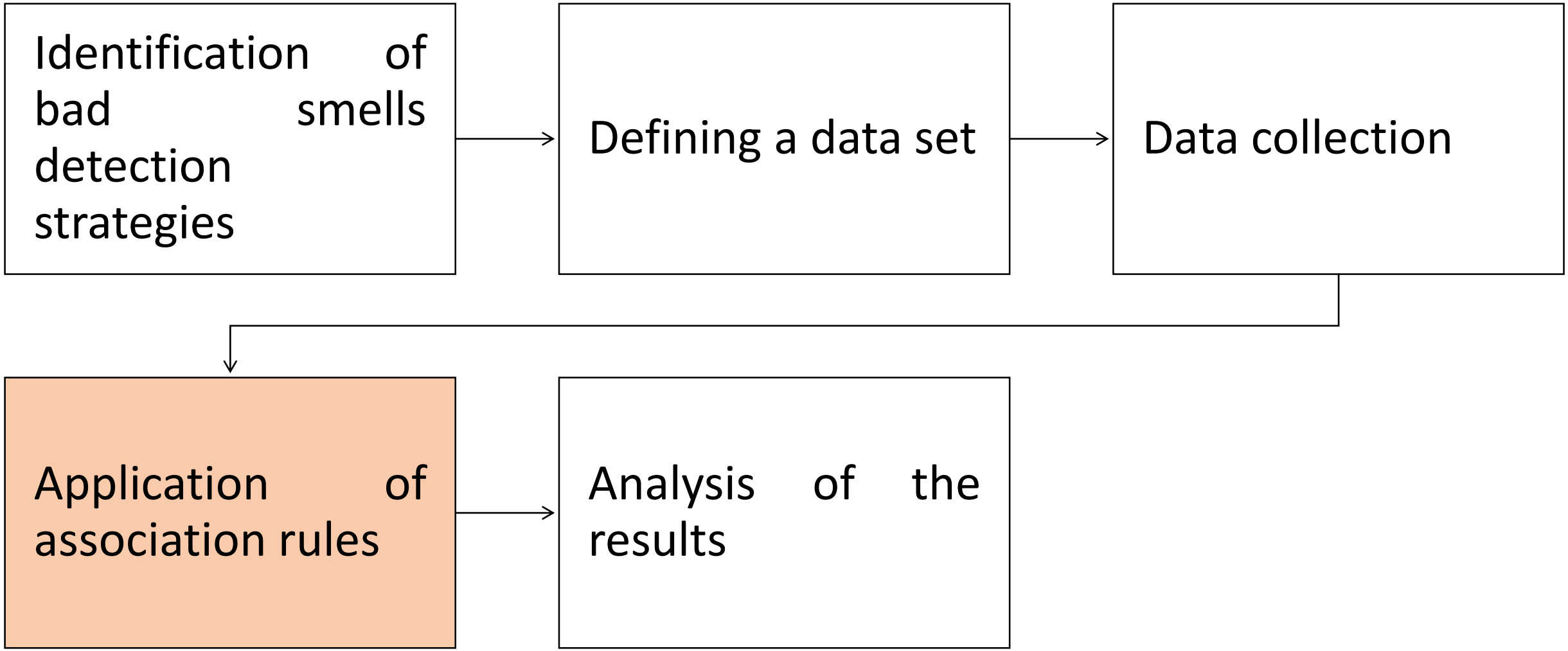
- Systems used in the data set
  - ✓Hibernate 4.2.0
  - ✓JHotDraw 7.5.1
  - ✓Kolmafia 17.3
  - ✓Webmail 0.7.10
  - ✓Weka 3.6.9

- Kolmafia doesn’t belong to Qualitas.class Corpus


Identification of bad smells detection strategies
Defining a data set
Data collection
Application of association rules
Analysis of the results

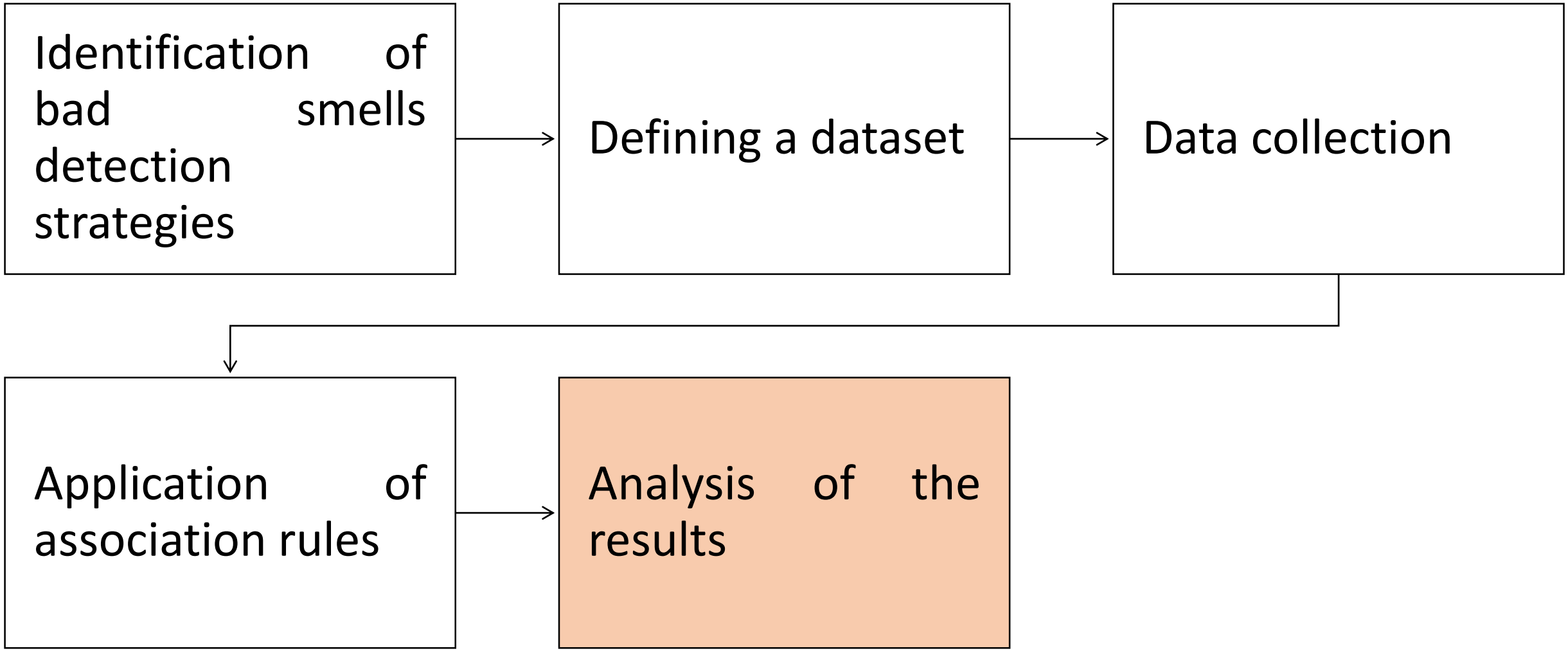
# Data Collection

- Metric
  - Qualitas.class Corpus
  - Eclipse 4.2 Juno
  - Metrics 1.3.6
- Design Pattern
  - Design Pattern Detection using Similarity Scoring
- Bad Smells
  - RAFTool
- Co-Occurrence
  - Design Pattern Smell


Identification of bad smells detection strategies
Defining a data set
Data collection
Application of association rules
Analysis of the results

# Application of Association Rules

- Combining items and extracting knowledge
- Rules
  - Support
  - Confidence
  - Conviction
- Concepts
  - Transaction
  - Antecedent
  - Consequent
- **Antecedent → Consequent**


Identification of bad smells detection strategies
Defining a dataset
Data collection
Application of association rules
Analysis of the results

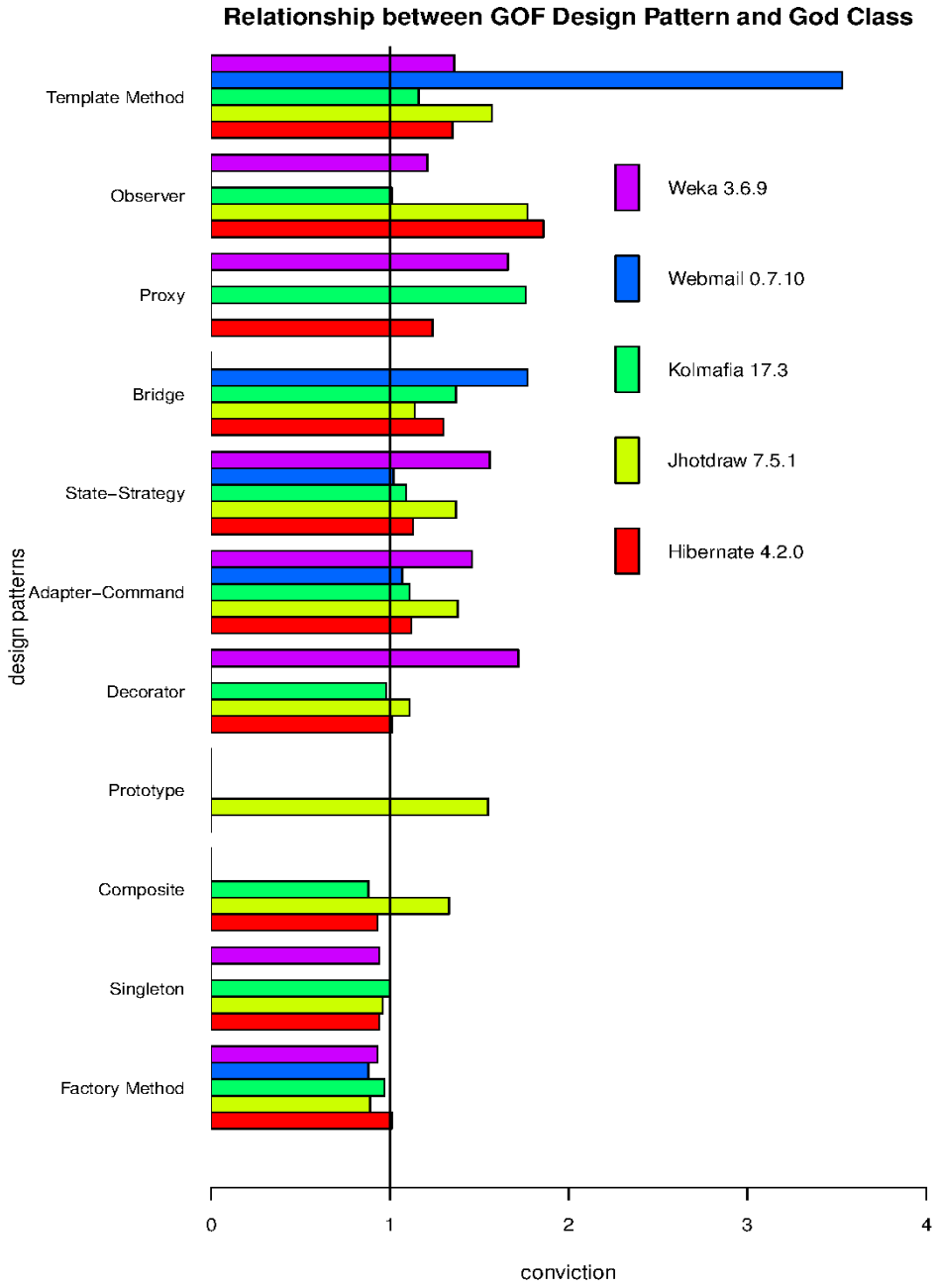
# Results

| Design Pattern | Hibernate | | JHotDraw | | Kolmafia | | Webmail | | Weka | |
|---|---|---|---|---|---|---|---|---|---|---|
| | I | II | I | II | I | II | I | II | I | II |
| Adapter-Command | 228 | 39 | 53 | 19 | 386 | 81 | 40 | 7 | 152 | 68 |
| Bridge | 56 | 16 | 40 | 9 | 14 | 5 | 6 | 3 | 0 | 0 |
| Composite | 12 | 0 | 12 | 4 | 8 | 0 | 0 | 0 | 0 | 0 |
| Decorator | 37 | 3 | 10 | 2 | 67 | 7 | 0 | 0 | 32 | 17 |
| Factory Method | 37 | 3 | 5 | 0 | 31 | 3 | 2 | 0 | 22 | 3 |
| Observer | 4 | 2 | 2 | 1 | 8 | 1 | 0 | 0 | 36 | 12 |
| Prototype | 0 | 0 | 21 | 9 | 0 | 0 | 0 | 0 | 0 | 0 |
| Proxy | 8 | 2 | 0 | 0 | 18 | 9 | 0 | 0 | 35 | 18 |
| Singleton | 232 | 3 | 13 | 1 | 77 | 9 | 1 | 1 | 34 | 5 |
| State-Strategy | 271 | 47 | 121 | 43 | 334 | 64 | 23 | 3 | 93 | 45 |
| Template Method | 87 | 27 | 16 | 7 | 54 | 13 | 4 | 3 | 22 | 9 |

God Class

I. Classes with design pattern

II. Classes with co-occurrence


Relationship between GOF Design Pattern and God Class
Template Method
Observer
Proxy
Bridge
State–Strategy
Adapter–Command
Decorator
Prototype
Composite
Singleton
Factory Method
design patterns
Weka 3.6.9
Webmail 0.7.10
Kolmafia 17.3
Jhotdraw 7.5.1
Hibernate 4.2.0
0
1
2
3
4
conviction

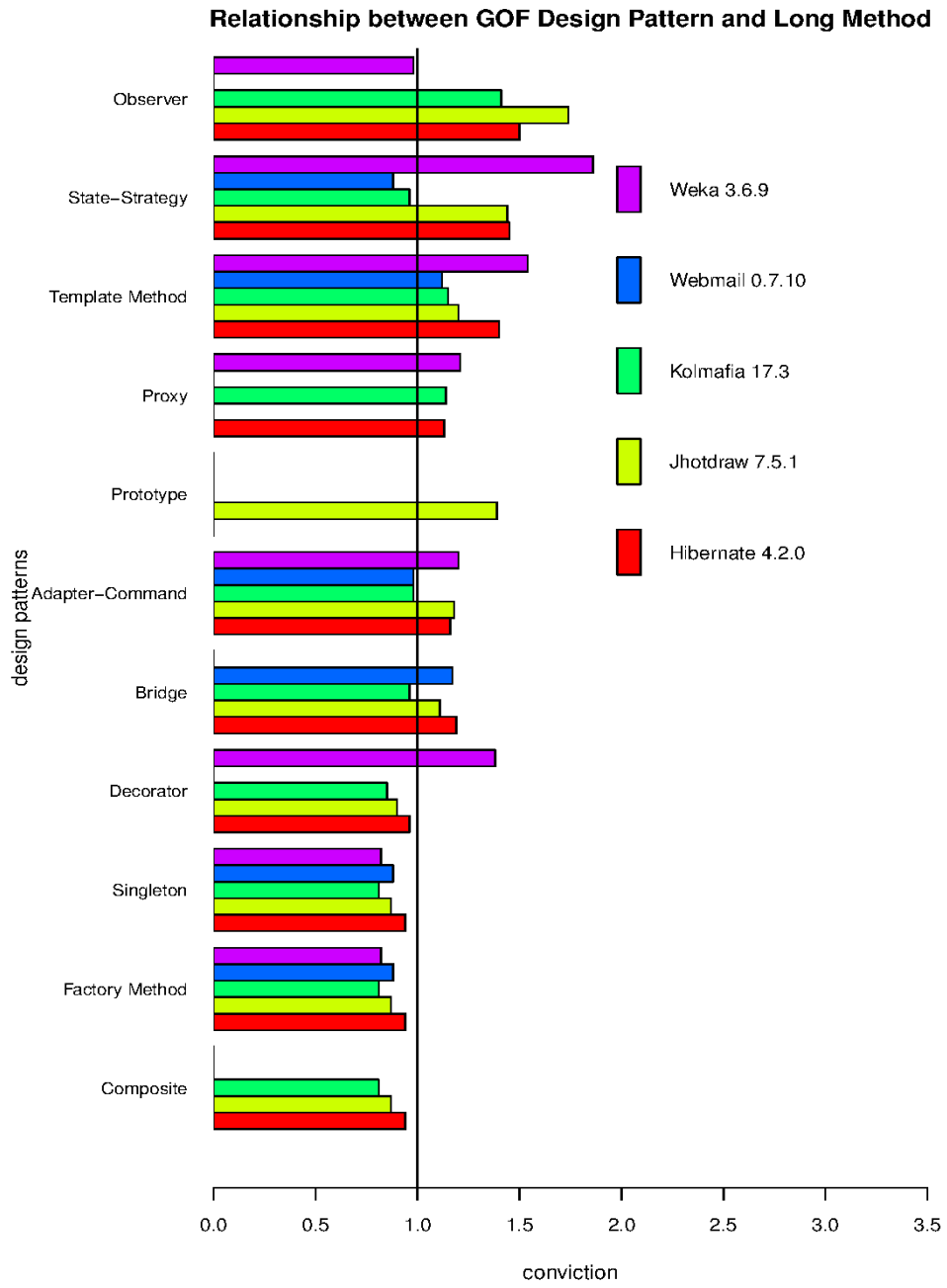
# Results

| Design Pattern | Hibernate | | JHotDraw | | Kolmafia | | Webmail | | Weka | |
|---|---|---|---|---|---|---|---|---|---|---|
| | I | II | I | II | I | II | I | II | I | II |
| Adapter-Command | 271 | 52 | 73 | 19 | 703 | 123 | 50 | 5 | 222 | 71 |
| Bridge | 61 | 13 | 51 | 11 | 19 | 3 | 8 | 2 | 0 | 0 |
| Composite | 8 | 0 | 29 | 0 | 37 | 0 | 0 | 0 | 0 | 0 |
| Decorator | 115 | 2 | 31 | 1 | 255 | 12 | 0 | 0 | 61 | 25 |
| Factory Method | 58 | 0 | 23 | 0 | 45 | 0 | 2 | 0 | 27 | 0 |
| Observer | 8 | 3 | 2 | 1 | 7 | 3 | 0 | 0 | 24 | 4 |
| Prototype | 0 | 0 | 16 | 6 | 0 | 0 | 0 | 0 | 0 | 0 |
| Proxy | 6 | 1 | 0 | 0 | 31 | 9 | 0 | 0 | 37 | 12 |
| Singleton | 340 | 0 | 15 | 0 | 672 | 0 | 1 | 0 | 83 | 0 |
| State-Strategy | 343 | 121 | 227 | 90 | 974 | 154 | 19 | 0 | 173 | 97 |
| Template Method | 275 | 90 | 47 | 13 | 161 | 48 | 14 | 3 | 34 | 16 |

Long Method

I. Methods with design pattern

II. Methods with co-occurrence


Relationship between GOF Design Pattern and Long Method
Observer
State–Strategy
Template Method
Proxy
Prototype
Adapter–Command
Bridge
Decorator
Singleton
Factory Method
Composite
design patterns
Weka 3.6.9
Webmail 0.7.10
Kolmafia 17.3
Jhotdraw 7.5.1
Hibernate 4.2.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
conviction

# RQ1. Do the design patterns defined in the GOF catalog avoid the occurrence of God Class and Long Method bad smells in software?

# Answer to RQ1...

- Some design patterns presented low co-occurrence
  - Composite
  - Factory Method
- Composite and Factory Method are design patterns intrinsically modular
- Singleton presented low co-occurrence with God Class
  - Low co-occurrence with God Class
  - Inconclusive in relation to Long Method
- Design Pattern not necessarily avoid occurrences of bad smells

**RQ2. Which design patterns of GOF catalog present co-occurrence with the God Class and Long Method bad smells?**

# Answer to RQ2...

- God Class
  - Template Method
  - Observer
  - Proxy

- Long Method
  - Observer
  - State-Strategy
  - Template Method

**RQ3. What are the more common situations in which the God Class and Long Method bad smells appear in software systems that apply GOF design patterns?**

# Answer to RQ3...

- Manual inspection
- Template Method ⟶ God Class
  - Several responsibilities assigned to the classes with template method
  - Poor planning and misapplication of Template Method
- Observer ⟶ Long Method
  - High code repetition
  - High responsibility assigned observer notification methods
  - Scattering and crosscutting concerns

# Threats to Validity

- The results can not be generalized
- Use of tools in data collection
- Manual Inspection

# Conclusion

- We studied co-occurrences between design pattern and bad smells
- We used a methodology based in software measurement
- The results pointed which design patterns not necessarily avoid the bad smells occurrences
- Co-occurrences identified
  - Template Method → God Class
  - Observer → Long Method
- Main situations
  - Poor planning and misapplication of design pattern
  - Scattering and crosscutting concerns
- Future Work
  - Investigate co-occurrences with others bad smells
  - Extend the system sample

# Thank you!

CONTACT US

Bruno L. Sousa

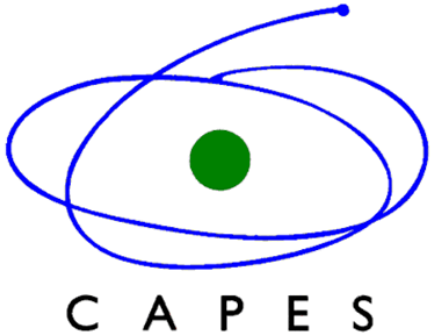
bruno.luan.sousa@dcc.ufmg.br

Mariza A. S. Bigonha

mariza@dcc.ufmg.br

Kecia A. M. Ferreira

kecia@decom.cefetmg.br





UFMG