Code Anomalies Flock Together

Exploring Code Anomaly Agglomerations for Locating Design Problems

MQS – May 22, 2019 – Paper presentation by

João Paulo de Freitas Diniz

The Paper

- Authors
  - PUC-RJ: 3
  - USC-Los Angeles: 1
- Venue:
  - International Conference on Software Engineering
    - ICSE 2016
- 41 citations (Google Scholar)

Overview

- Introduction
- Background
- Study definition
- Results and Analysis
- Threats to validity

Some definitions

- Code anomalies:
  - Bad Smells
- Components
  - Sub-systems
  - Packages
- Code elements
  - Classes, interfaces, methods, constructors, fields

Design problems

- Structures that indicate **violation** of
  - Intended design **rules**, OR
  - Fundamental design **principles**

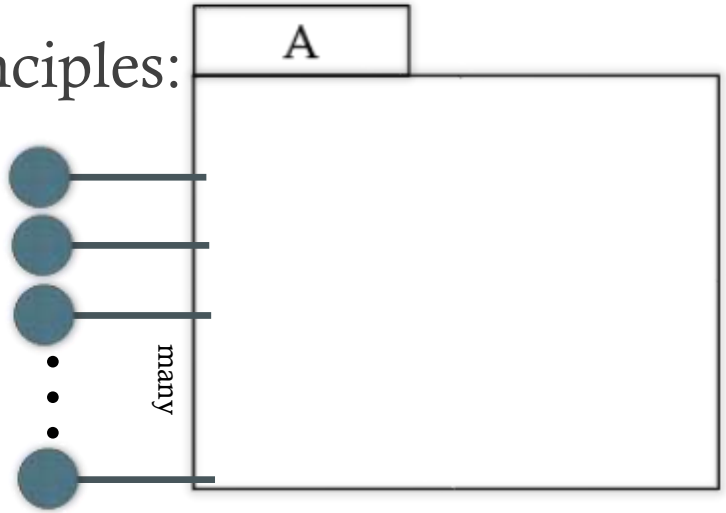
- Often affect **multiple** elements
- Are **hard** to identify in the source code
- **Negatively** impact design **quality**

Design problems present

- Fat Interface
- Unwanted Dependency
- Component Overload
- Cyclic Dependency
- Delegating Abstraction
- Scattered Concern
- Overused Interface
- Unused Abstraction

Fat Interface

- General and ambiguous **entry point** of a design component (sub-system/package)
  - provides **non-cohesive** services
  - complicates the **logic** of its clients
- **Violates** the (well-known) principles:
  - Cohesion
  - Abstraction
  - Separation of concerns



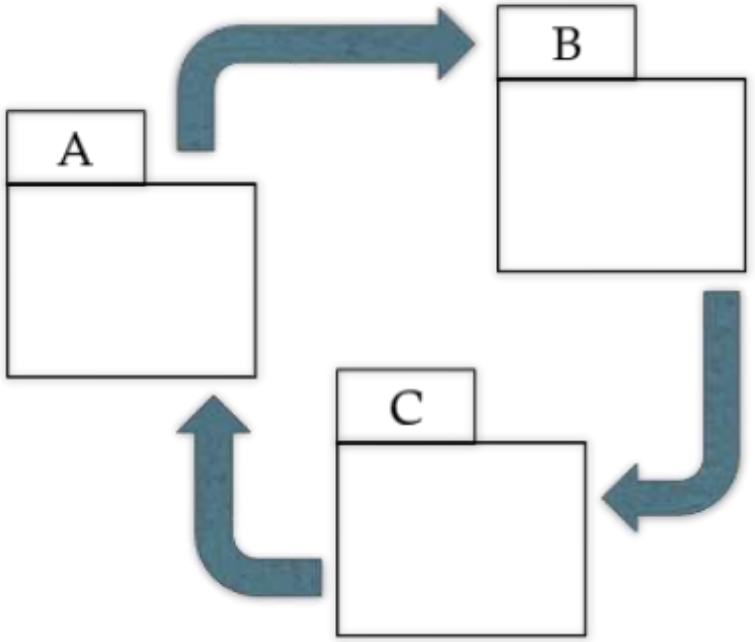
Unwanted Dependency

- Violation of an intended design rule.

- Example:
  - Two components should not communicate

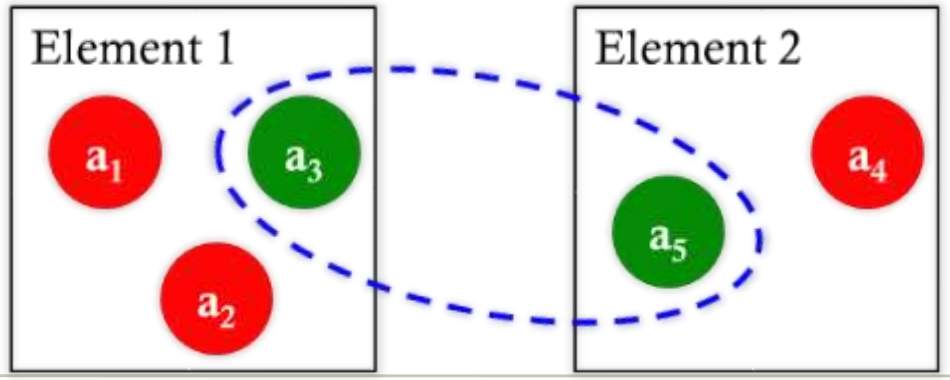
Cyclic Dependency

- Two or more components that
  - directly or indirectly
  - depend on each other

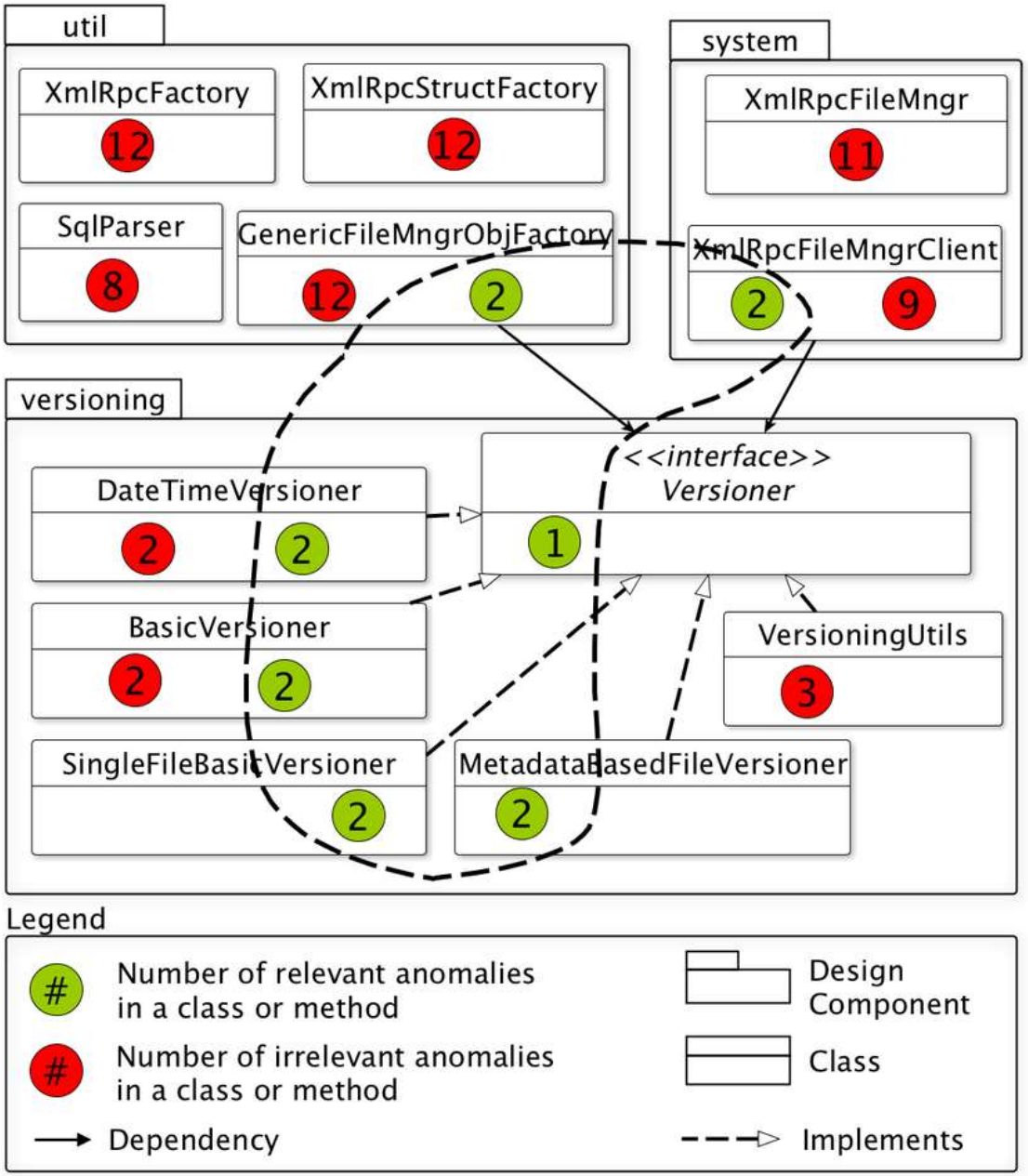


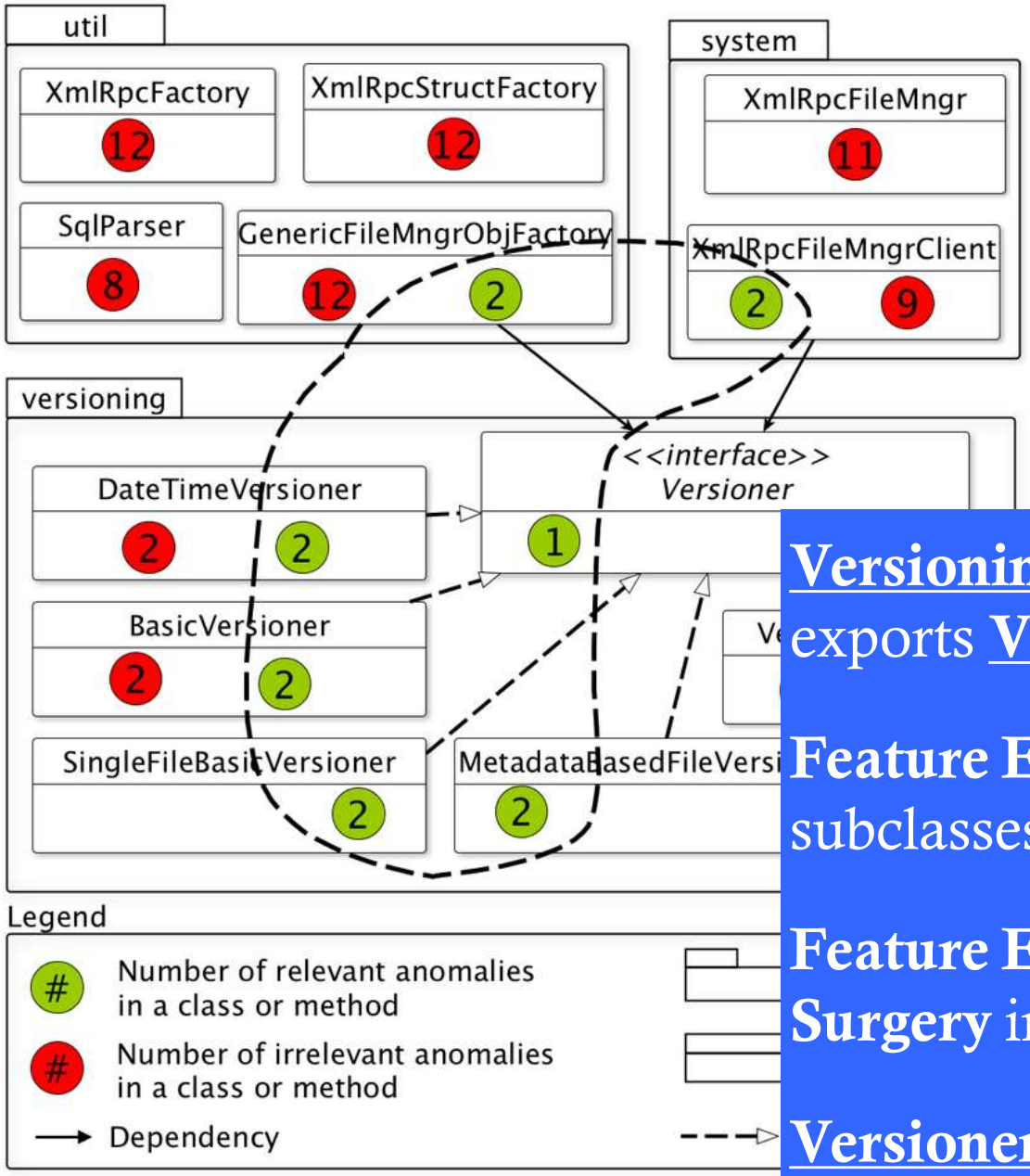
Code anomaly agglomerations

- Groups of **inter-related** code anomalies
- Two code anomalies are **related** if
  - their host program elements are **connected through**
    - Method calls
    - Inheritance



Motivating Example





Versioning component exports **Versioner** interface.

Feature Envy in all of its subclasses.

Feature Envy and **Shotgun Surgery** in its clients.

Versioner is a **Fat Interface**.

Study definition

Research questions

1. RQ1: Are **syntactic** anomaly agglomerations **sufficient indicators** of design problems?

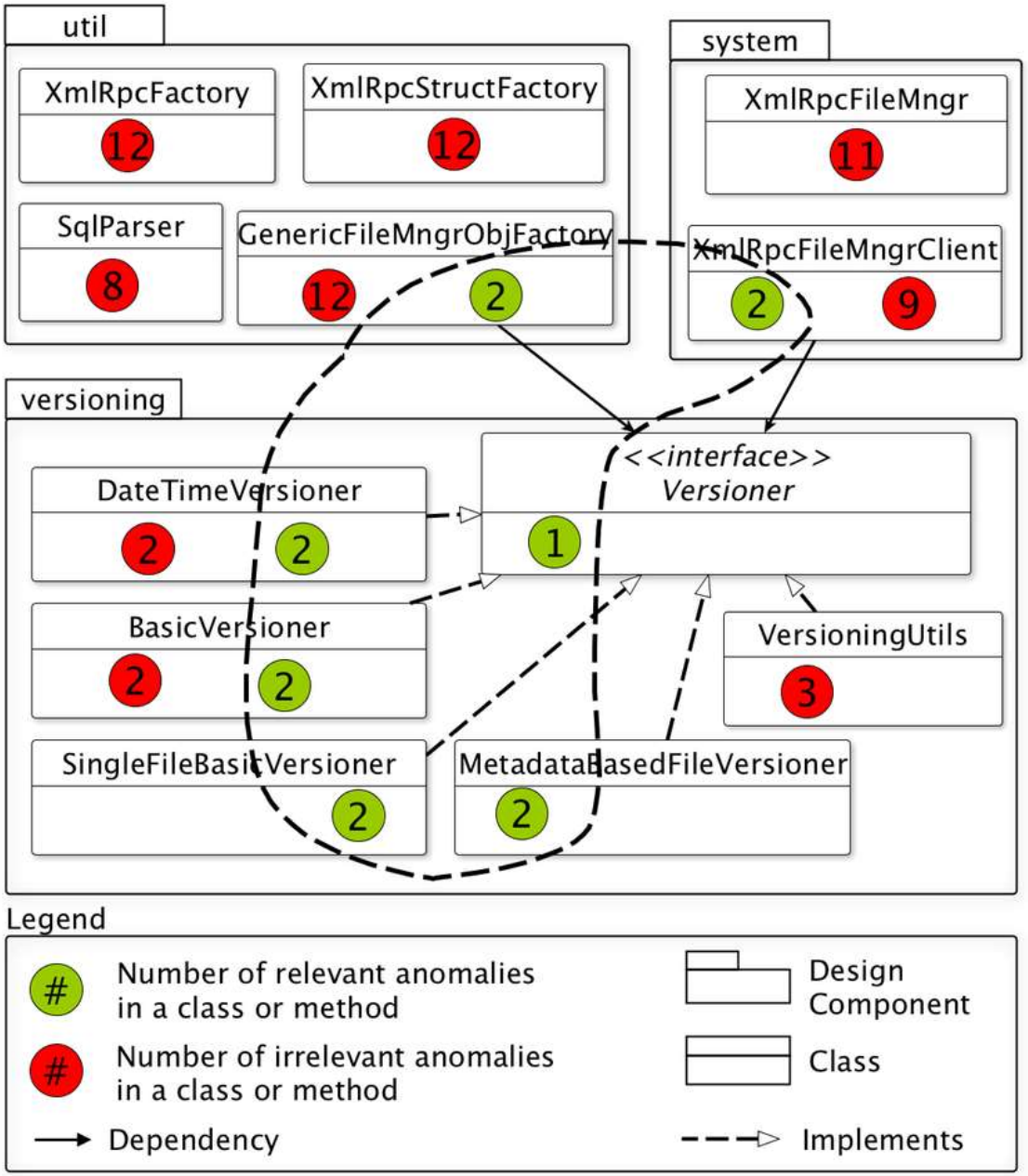
2. Q2: What **proportion** of design problems manifest as anomaly agglomerations in **early** versions of a program?

Syntactical agglomerations

- Inter-component agglomeration

- Intra-component agglomeration

Syntactical agglomerations: inter and intra-components



Semantic agglomerations

- Anomalous code elements **realizing** a single concern
- **Not modularized** by design components
- **Domain-specific** concerns

Photo Label Management concern

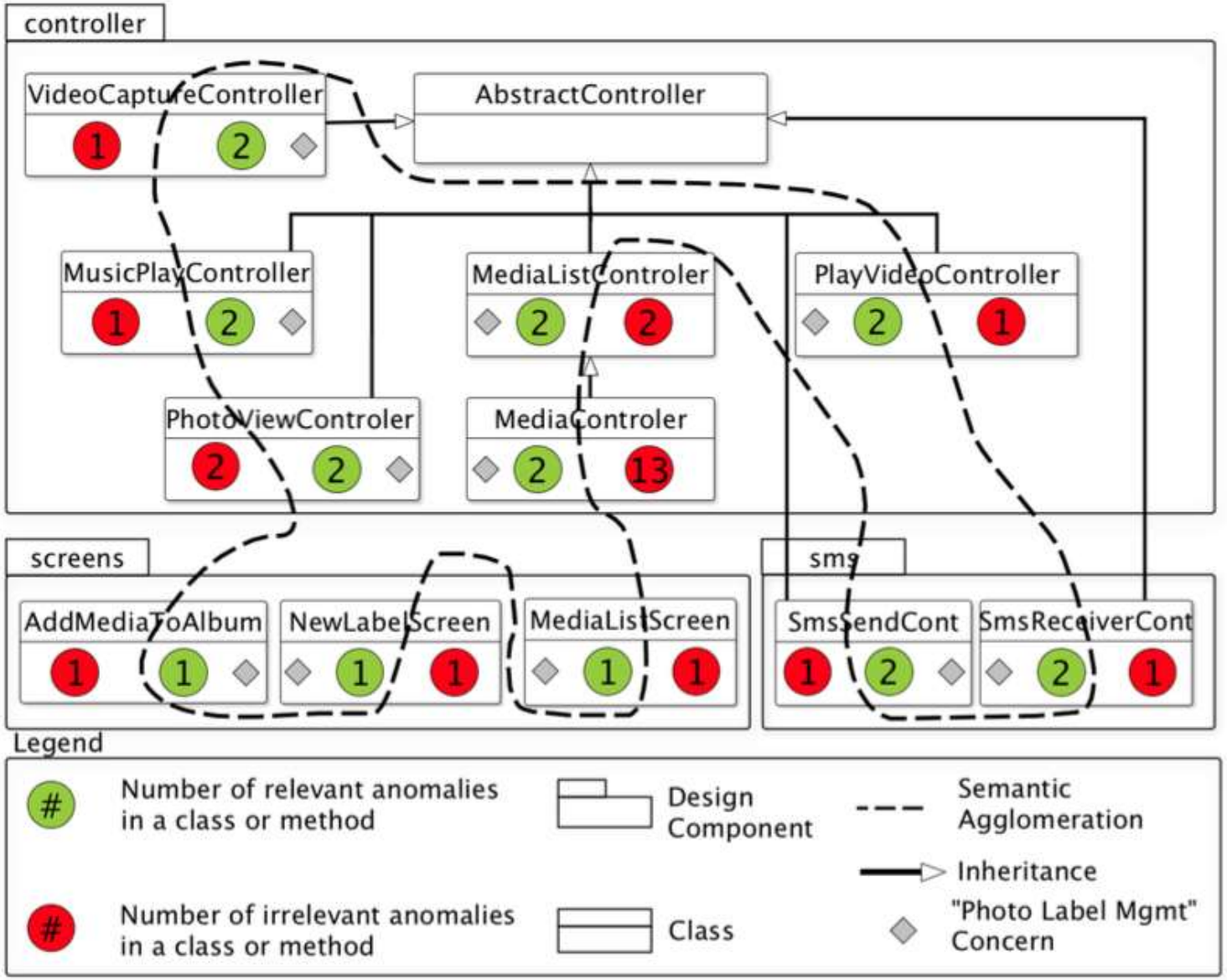
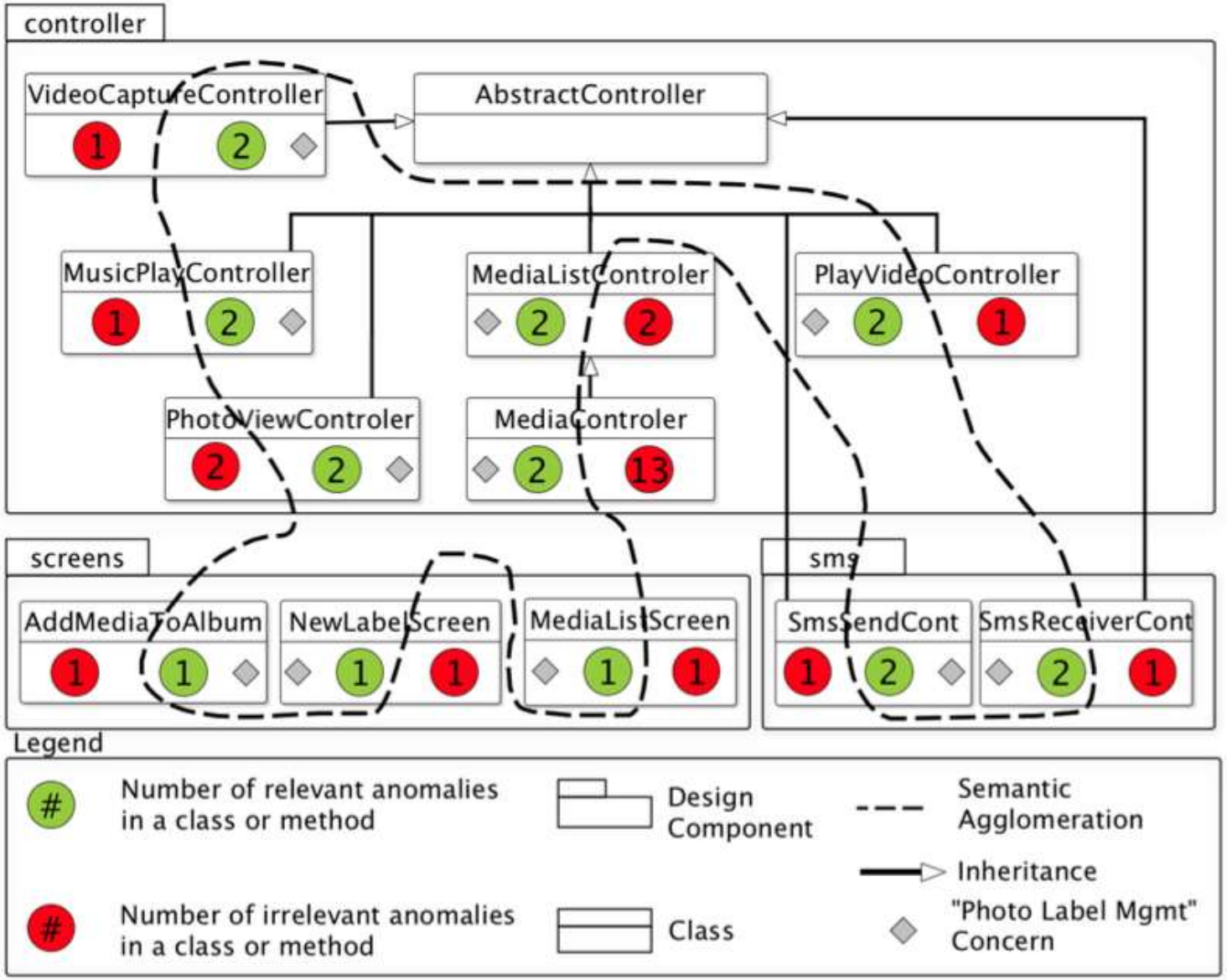


Photo Label Management
semantic agglomeration



Scattered concern with many **Divergent Change** and **Shotgun Surgery anomalies**

Target systems

System	Type	Design	KSLOC
HW	Web Framework	Layers	8
MM	Software Product Line	MVC	10
S1	Desktop Application	Client-Server	122
S2	Desktop Application	Client-Server	118
S3	Desktop Application	Client-Server	93
S4	Web Application	MVC	116
OODT	Middleware	Layers	129

Health Watcher, Mobile Media, Apache OODT and 4 proprietary

Data collection

procedure

Identifying design problems

1. Initial **list** provided by **original developers** through a **questionnaire**:
   1. Explanation
   2. Relevance
   3. Related maintenance effort
   4. Which elements contributes to each design problem
2. Additional identification
   1. Using source code and system design (documentation)
   2. Suite of design recovery tools (from a previous thesis)

Identifying design problems

Design problem	Instances
Fat Interface	114
Unwanted Dependency	2,145
Component Overload	141
Cyclic Dependency	351
Delegating Abstraction	35
Scattered Concern	216
Overused Interface	39
Unused Abstraction	59

Concern mappings

1. Concern collection:
    1. System's **developers** provided a list
    2. Mallet tool provided another list
        - Concern location tool
            - Identify each code element realizing each concern
2. Authors **computed** and **compared** two sets of semantic agglomerations
    - based on both sources above

Code anomalies

Code anomalies

- Data Class
- Divergent Change
- God Class
- Shotgun Surgery
- Feature Envy
- Long Method
- Long Parameter List

Detection

- Well-known metrics-based strategies
- Used in 4 previous empirical studies
- Precision > 80%

Agglomerations

Tool named **Organic**, which implements:

- 7 detection strategies for code anomalies

- 2 algorithms for computing agglomerations:
  - Syntactically
  - Semantically

  (pseudo-code provided)

Correlating **agglomerations** and **design problems**

- **Individual** code anomalies
  - A design problem is **related** to an individual code anomaly if the DP is realized (fully or partially) by the code element affected by the anomaly.

- **Agglomerations** of code anomalies
  - A design problem and an agglomeration are related if **they co-occur** in at least two code elements

Results

and Analysis

Exploring agglomerations

Agglomeration	AG-DP	NoAG-NoDP	p-value	ORs
Intra-component	247	3996	<0.001	5,669636
Inter-component	167	4207	<0.001	4,878982
Syntactic	296	3759	<0.001	5,513531
Semantic	97	4463	<0.001	7,392637
All	**312**	**4596**	**<0.001**	**2,999686**

Reducing search space for DPs

Amount of agglomerations related to design problems

Detached anomalies irrelevant to locate design problems

Agglomeration	AG-DP	NoAG-NoDP	p-value	ORs
Intra-component	247	3996	<0.001	5,669636
Inter-component	167	4207	<0.001	4,878982
Syntactic	296	3759	<0.001	5,513531
Semantic	97	4463	<0.001	7,392637
All	**312**	**4596**	**<0.001**	**2,999686**

Reducing search space for DPs

Amount of agglomerations related to design problems

Detached anomalies irrelevant to locate design problems

Agglomeration	AG-DP	NoAG-NoDP	p-value	ORs
Intra-component	247	3996	<0.001	5,669636
Inter-component	167	07	<0.001	4,878982
Syntactic	296	37	<0.001	5,513531
Semantic	97	4463	<0.001	7,392637
All	**312**	**4596**	**<0.001**	**2,999686**

Statistical significance

Fisher's exact test
Confidence level
of 99%

Agglomeration	AG-DP	NoAG-NoDP	p-value	ORs
Intra-component	247	3996	<0.001	5,669636
Inter-component	167	4207	<0.001	4,878982
Syntactic	296	3759	<0.001	5,513531
Semantic	97	4463	<0.001	7,392637
All	**312**	**4596**	**<0.001**	**2,999686**

High correlation between agglomerations and design problems in the 7 target systems.

Strength of the relation

Odds Ratio

Agglomeration	AG-DP	NoAG-NoDP	p-value	ORs
Intra-component	247	3996	<0.001	5,669636
Inter-component	167	4207	<0.001	4,878982
Syntactic	296	3759	<0.001	5,513531
Semantic	97	4463	<0.001	7,392637
All	**312**	**4596**	**<0.001**	**2,999686**

Strength of the relation

Agglomeration	AG-DP	NoAG-NoDP	p-value	ORs
Intra-component	247	3996	<0.001	5,669636
Inter-component	167	4207	<0.001	4,878982
Syntactic	296	3759	<0.001	5,513531
Semantic	97	4463	<0.001	7,392637
All	**312**	**4596**	**<0.001**	**2,999686**

The chance of each anomaly **within** a syntactic agglomeration being related to a design problem is **5,51 times higher** than each detached anomaly.

Syntactical agglomerations

Intra and inter-component agglomerations are **complementary**

Agglomeration	AG-DP	NoAG-NoDP	p-value	ORs
Intra-component	247	3996	<0.001	5,669636
Inter-component	167	4207	<0.001	4,878982
Syntactic	296	3759	<0.001	5,513531
Semantic	97	4463	<0.001	7,392637
All	**312**	**4596**	**<0.001**	**2,999686**

Intra-component $\cup$ Inter-component = 296.

RQ1

Are **syntactic** anomaly agglomerations
sufficient indicators of design problems?

Semantic agglomerations

Syntactic and Semantic agglomerations are **complementary**

Agglomeration	AG-DP	NoAG-NoDP	p-value	ORs
Intra-component	247	3996	<0.001	5,669636
Inter-component	167	4207	<0.001	4,878982
Syntactic	296	3759	<0.001	5,513531
Semantic	97	4463	<0.001	7,392637
All	**312**	**4596**	**<0.001**	**2,999686**

Syntactic ∪ Semantic = 312.

RQ2

What **proportion** of design problems manifest as anomaly agglomerations in **early** versions of a program?

Agglomerations as congenital problems?

- For each system, the authors computed:
  - The number of design problems in the first versions
  - The proportion of design problems related to agglomerations
- **Considerable** number of design problems was introduced in the **first** version
- High correlation between design problems and agglomerations
- They expected that most of the problems were evolutionary

Agglomerations as congenital problems?

- They expected that most of the problems were evolutionary
  - **But not!**
  - 75% for MM, S1, S2 and S4
  - ~67% for the other
- 23% of the agglomerations of the 1st version of MM are still present in the last version
- For HW, 39%

Another analysis

Table 4: Relation of Agglomerations and Design Problems: Precision and Recall

Agglom.	Intra-component			Inter-component			Syntactic			Semantic		
System	P	R	DP	P	R	DP	P	R	DP (*)	P	R	DP (**)
OODT	62%	30%	196	73%	83%	549	70%	97%	640 (535)	91%	65%	431 (201)
MM	42%	31%	12	82%	23%	9	57%	55%	21 (21)	100%	13%	5 (3)
HW	39%	11%	22	45%	87%	163	44%	87%	163 (141)	75%	100%	187 (24)
S1	43%	23%	64	51%	21%	58	47%	37%	104 (86)	81%	12%	34 (18)
S2	38%	9%	77	40%	7%	60	39%	13%	110 (83)	82%	5%	47 (21)
S3	38%	9%	76	39%	6%	58	38%	12%	109 (84)	75%	6%	50 (25)
S4	66%	89%	66	34%	60%	45	49%	93%	69 (27)	100%	13%	10 (6)
Median	**42%**	**23%**		**45%**	**23%**		**47%**	**55%**		**82%**	**13%**	
SD	**0.118**	**0.281**		**0.182**	**0.349**		**0.109**	**0.368**		**0.107**	**0.370**	

* Number of design problems exclusively related either to intra-component or to inter-component agglomerations.
** Number of design problems exclusively related to semantic agglomerations.

Threats to validity

- Construct
  - Errors in the identification of DPs and CAs
- Conclusion
  - Small number of the evaluated versions
- Internal and External
  - Degree to which the findings can be generalized

Questions

???