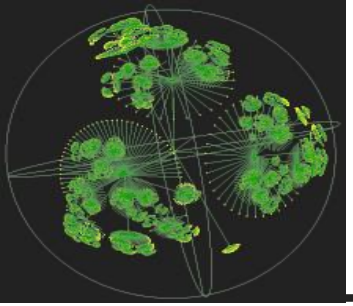


ESSeRE

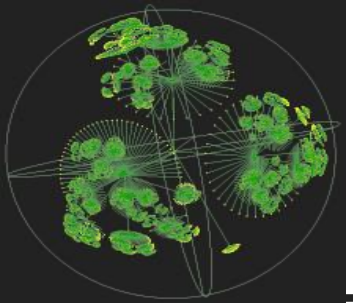
Evolution of Software Systems and Reverse Engineering

UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA

Towards assessing software architecture quality by exploiting code smell relations

Francesca Arcelli Fontana, Vincenzo Ferme, Marco Zanoni
University of Milano Bicocca, Italy
University of Lugano, Switzerland

Software Architecture and Metrics (**SAM 2015**)
ICSE 2015, Firenze, May 16, 2015



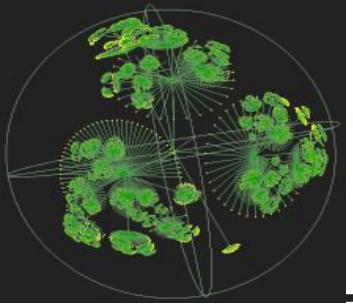
Problem Statment

There is no standard set of base measures to assess software architecture quality [Nord et al. 2014].

We can consider metrics and other different anomalies as code smells (CS), structural antipattern, architecture smells (AS), ...

Could some code anomalies, as code smells, and recurring patterns of them, be key indicators of **architecture degradation**?

Cluster of smells have more effect on maintainabilty than isolated smells.



Our analysis

Detected six smells: **God Class, Data Class, Brain Method, Shotgun Surgery, Message Chain, Dispersed Coupling**

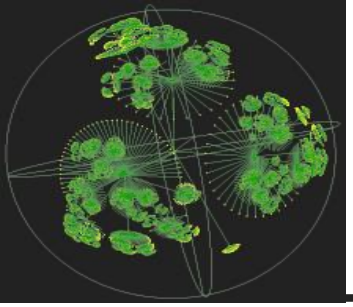
Detected on 74 systems of the Qualitas Corpus (N.Packages: 3.420, N° Classes: 51.826)

Code smell relations:
Contained smells: for every code smell, the set of all other smells contained in the same class/method, e.g., a Brain Method contained in a God Class;

Used smells: for God Class, the set of Data Classes it uses (accessed data);

Calling smells: for God Class, Data Class, Brain Method, Shotgun Surgery, the set of other smelly classes/methods calling the considered smell;

Called smells: for Dispersed Coupling, the set of other smelly classes/methods that are called by the smelly method.



Our finding

Frequency of smells (which smells are more frequently found in software)**:**

- **Data Class** is present in 96% of the systems
- **Brain Method** and **Dispersed Coupling in** more than 90%
- **God Class** in more than 82%
- **Message Chains** more than 50%

Related Code Smells :

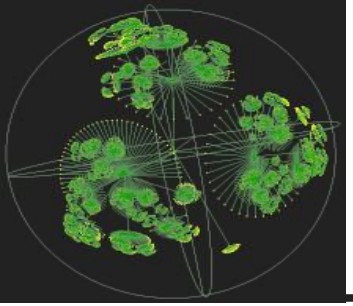
26% of God Classes use at least a Data Class.
53% of Shotgun Surgery are called (at least) from one class affected by a smell
58% of God Class are called by at least one class with CS
58% of God Classes contain other smells.
70% of classes called by Dispersed Coupling are affected by CS.

Co-Occurred Code Smells:

- 3% Brain Method and Dispersed Coupling
- 3% Brain Method and Message Chains

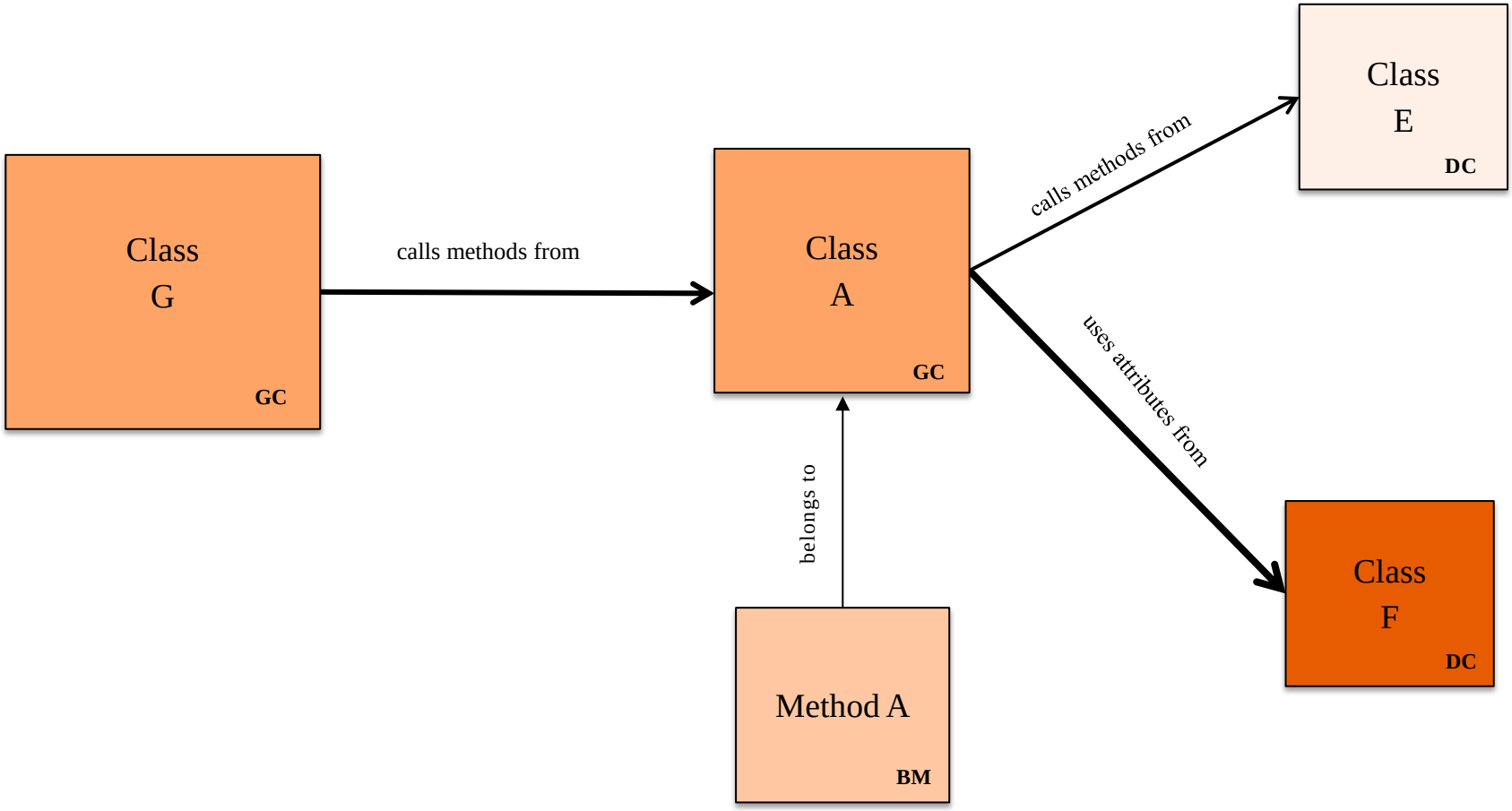
Code Smell Relations View

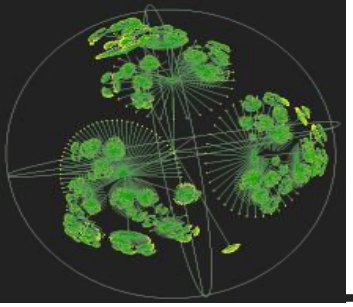
Useful to investigate potential problems pointed out by the presence of related CS



Code Smell Relations View

UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA





Our Proposal

We would like to exploit the information on code smells relations/correlations towards the detection of possible architectural anomalies.

→ extend our set of CS detection rules, consider other CS relations, identify other patterns of smells, manually check if we find some AS (known/new AS).

→ exploit an existing tool, as inFusion, to detect both CS and AS, statistically analyze if the correlations among CS lead to some AS

If we find interesting correlations, we can suggest the **prioritization of the smells** to be removed → Ranking of code smells.

Knowing the simultaneous occurrence of CS, we can **improve the refactoring process** (our Code Smell Relations View)

What can we use to assess software architectures? And improve them?
Metrics → Code Smells → Architectural Anomalies →…and/or History…Views….Technical Debt Index ??? Which practical **suggestions** can I get to **improve** the architecture?