OPUS Research Group

PRIORITIZING SOFTWARE ANOMALIES WITH METRICS AND ARCHITECTURE BLUEPRINTS

Everton Guimaraes
Alessandro Garcia
Eduardo Figueiredo
Yuanfang Cai

Introduction

- The progressive insertion of software anomalies
  - Architectural Problems[1,2,3]
  - Architecture degradation[4]
- Most part of architecture relevant anomalies can not be detected only by source code analysis
- It is essential to distinguish which **code anomalies**
  - … have impact on **software architecture**
  - … should be **prioritized** and removed, so its propagation during the system evolution can be avoided.

1. Macia, J. Garcia, D. Popescu, A. Garcia, N. Medvidovic and A. von Staa. Are Automatically-Detected Code Anomalies Relevant to Architectural Modularity? . In Proc. of 11th AOSD, pp. 167-178, USA, March 2012.
2. I. Macia, R. Arcoverde, A. Garcia, C. Chavez and A. von Staa. On the Relevance of Code Anomalies for Identifying Architecture Degradation Symptoms. In Proc. of 16th CSMR, Szeged, Hungary, March 2012.
3. J. Garcia, D. Popescu, G. Edwards and N. Medvidovic, Identifying Architectural Bad Smells. In Proc. of 13th CSMR, March 2009.
4. L. Hochstein and M. Lindvall. Combating Architectural degenerations: A Survey. Information and Software Technology, Vol. 47, Issue 10, pp. 643-656, July 2005.

Context

- Detection Strategies
  - Metrics are the most popular artifact to detect severe anomalies
  - Developers can defined their own strategies (e.g. thresholds)
  - Most part of detection strategies in the state-of-art fail to assist developers in prioritizing severe anomalies
  - Other limitations
    - The metrics alone are often agnostic to the architecture structure
    - Developers tend to consider that all measures and architecture components have the same relevance.

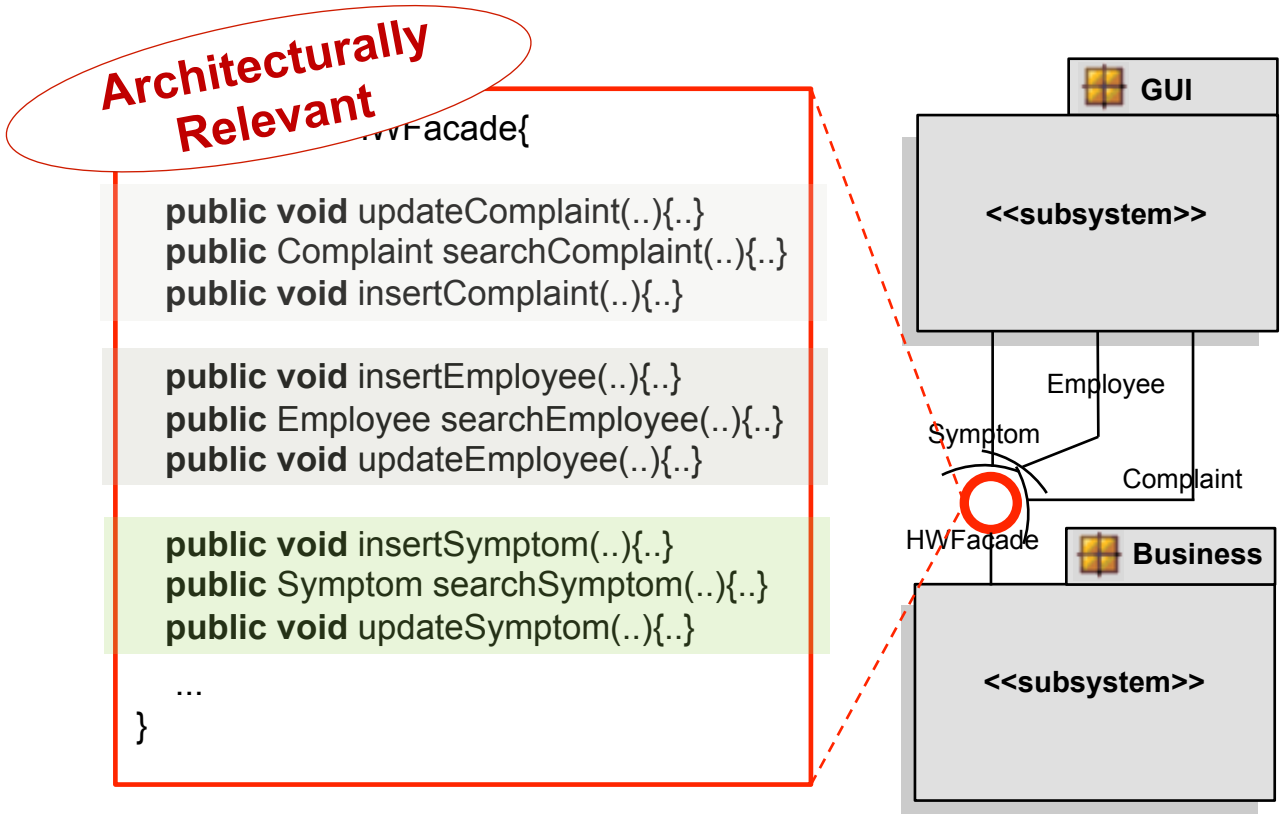
Context

- On the other hand…
  - Architecture design models are often of informal nature
  - Represent **architecture blueprints**
    - **Blueprints** are used for communication purposes.
    - **Architecture blueprints** are omnipresent in many software projects
    - However, it is unfeasible the collection of measures from them
- Our question is..
  - “to what extent the use of architecture blueprints would enhance the prioritization of architecture relevant anomalies?”

Software Anomaly

- Software Anomaly is a situation that suggests a potential problem on software structure
  - Code Anomalies
  - Design Anomalies (e.g. architectural drift)
- Some anomalies..
  - … can be observed in other artifacts (e.g. architecture blueprints)
  - … while other anomalies can only be observed looking at the source code.
- Also, studies revealed that…
  - Software anomalies are responsible for undesirable modifications
    - Ex1. Source code structure affects by anomalies is change proneness.
    - Ex2. code Anomalies also favor the occurrence of faults.

Relevance of Code Anomalies



Architecture Blueprints

- Architecture design blueprints..
  - … have been exploited in many different software engineering activities (e.g. model transformation optimization)
- In this sense, we are investigating
  - how **architecture blueprints** would enhance the prioritization process.
  - What information is useful to be showed in the **architecture design blueprint**
    - For example: dependencies strength between components.
    - Additional information to complement the information provided by the metrics and source-code.

Study Methodology

- Research Question
  - *How can architecture blueprints help the prioritization of relevant code anomalies?*
- Hypothesis
  - **H1:** *The use of **architecture blueprints** as additional artifact to detect anomalies do not provide any enhancement on precision measures.*
  - **H2:** *The use of **architecture blueprints** to improve the code anomaly detection process do not impact on recall measures.*

Study Methodology

- Target Application
  - Mobile Media SPL
  - Code Smells Reference List
- Experimental Procedures
  - Subjects were organized into 2 groups:
    - BP: group provided with code artifacts + architecture blueprints
    - NBP: groups provided only with code artifacts
  - Documentation describing Mobile Media
- Each group of subjects…
  - …should reason about the system information, architecture blueprints and software metrics.
  - … identify which classes could be candidates to present code anomalies.

Controlled Experiment

- Our goal was to compare
  - “the efficiency of detecting code anomalies with and without the use of architecture blueprints”
- So, we have used **Precision** and **Recall** measures.
  - These two metrics leverage to other three metrics.
    - True Positives
    - False Positives
    - False Negatives

$$Precision = TP/(TP+FP)$$

$$Recall = TP/(TP+FN)$$

Controlled Experiment

- It is important to say that…
  - A high precision implies that identified more relevant anomalies than the irrelevant ones.
  - A high recall implies that a subject identified most of the relevant anomalies.

Measure	Anomaly	N		Mean (%)		S.D.	
		BP	NPB	BP	NPB	BP	NPB
Precision	DC	10	24	47.9	43.6	26.7	27.1
	GC	14	10	50.9	66.8	25.5	23.6
	SS	11	20	25.8	21.5	5.4	19.8
Recall	DC	10	24	44.5	39.1	16.7	28.4
	GC	14	10	82.1	73.3	24.9	30.8
	SS	11	20	33.0	21.3	0	24.3

Controlled Experiment

- Architecture Design Blueprints and Precision
  - Precision has increased for 2 out of the three anomalies analyzed
    - Shotgun Surgery (4%)
    - Divergent Change (4.3%)
  - However, precision was not improved for God Class anomaly
    - Difference between groups was 16%.
    - Reasons:
      - It is a more intuitive anomaly than the other two
      - Some subjects have not followed the inspection process correctly
        - Misinterpretation of metrics values -> lead to a high number of False Positives.

Measure	Group	N	Mean (%)	Calc. p-value
Precision	BP	35	46.2	0.100
	NBP	55	39.7	

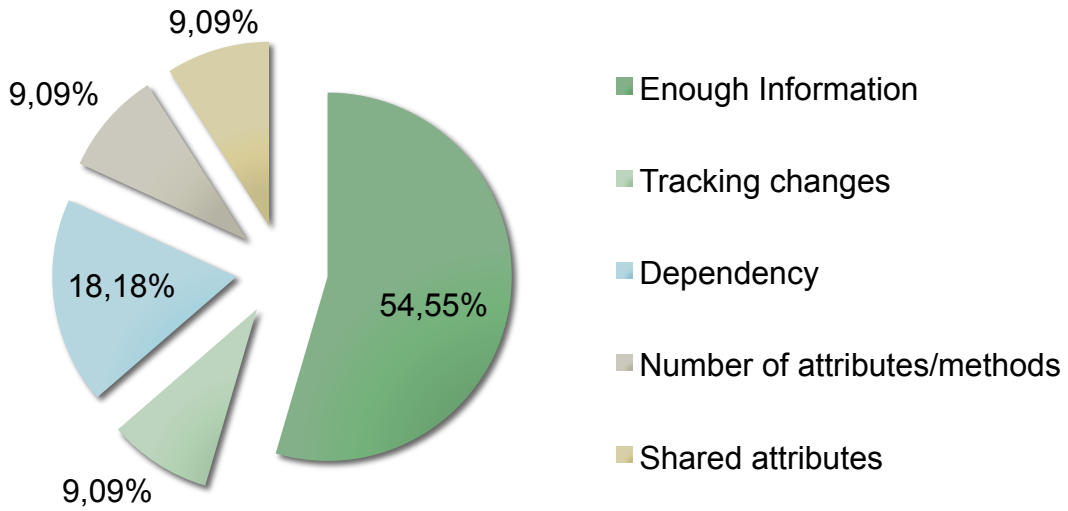
Controlled Experiment

- Architecture Design Blueprints and Precision
  - Recall measures increase for the BP group
    - Shotgun surgery (30%), Divergent Change (5.3%), God Class (8.8%)
    - Average recall measures increased by 21%
  - The use of blueprints..
    - Improved the effectiveness of anomaly detection
    - Decrease the number of False Negatives
  - The Lower the number of False Negatives, the higher is the Recall measures

Measure	Group	N	Mean (%)	Calc. p-value
Recall	BP	35	62.3	0.001
	NBP	55	41.3	

Controlled Experiment

- Usefulness of Design Blueprints
  - Around 71.4% of subjects judged the architectural blueprints useful on the detection and process
  - From this set of subjects…
    - We asked them to identify other information that would be helpful for detecting and prioritizing relevant code anomalies.



Final Remarks and Future Works

- Our findings..
  - The use of design blueprints has (somehow) improved precision and recall measures.
  - Feedback from Subjects
    - Some of them has not correctly followed the inspection sequence
    - lead the detection of more False Positives
- Hy… Rejected
  - H1 – we observed that the design blueprints can somehow improve precision measures.
    - statistical tests indicate cannot be considered as being …ignificant Accepted
  - H2 - We observed that the recall measures were affected by the used of design blueprints on the detection process.
    - This hypothesis can be confirmed with an acceptable statistical significance.

Final Remarks and Future Works

- Although this study was conducted with undergrad and graduate students…
  - It is not a limitation because it a first investigation on how to improve detection/prioritization of relevant code anomalies
- As a future work, we intend to:
  - Investigate more **software anomalies** that have been considered relevant for software architecture.
  - Investigate what are the main **characteristics** of classes that led to **False Positives** and **False negatives**
  - Provide **architecture blueprint** tailored with other substantial information

OPUS Research Group

Thank You!

QUESTIONS?

Everton Guimaraes
Alessandro Garcia
Eduardo Figueiredo
Yuanfang Cai