Towards Improving Software Architecture Degradation Mitigation by Machine Learning

Sebastian Herold
Christoph Knieke
Mirco Schindler
Andreas Rausch

Karlstad University, Sweden
Clausthal University of Technology, Germany

ADAPTIVE ESES 2020
29th October, 2020

Contact: sebastian.herold@kau.se
About the presenter

• Short resume
  • 2015-: Associate Professor in Computer Science at Karlstad University
  • 2014-2015: Research Fellow at Lero – The Irish Software Research Centre
  • 2011-2014: Post-doctoral researcher at Clausthal University of Technology

• Research Interests
  • Software Architecture, in particular degradation
  • Software Evolution and Modernization
What is this about?

Software Architecture Degradation

“The continuous divergence between the intended (prescriptive) and the as-implemented (descriptive) architecture.”

Machine Learning

“Software learning to perform a certain class of task better over time based on previous experience.”
Motivation

• Typical activities in architecture degradation mitigation
  • Architecture recovery
  • Consistency checking / degradation detection
  • Degradation analysis / comprehension
  • Degradation repairing

• All activities are labour-intensive and intellectually challenging

Motivating question:

How can we make use of machine learning to mitigate architecture degradation more efficiently?
Core of the approach

- **Architecture Recovery and Consistency**
  - Recover intended architecture from code
  - Check consistency between intended architecture and code

- **Degradation Cause Analysis**
  - Identify the reasons for/causes of architectural inconsistencies, identify actual violations

- **Recommending Repair Actions**
  - Recommend refactoring of implementation or architectural adaptations to resolve degradation
ML in Architecture Recovery & Consistency

• Architecture Recovery often understood as: clustering implementation elements into architectural components
• Clustering a typical task in unsupervised learning

Main idea: express architectural concepts in terms of code-based features

• There are more complex architectural concepts like patterns and guidelines
• Architectural decisions affect more than the system decomposition
• Result of recovery is most often not the intended but the implemented architecture
ML in Degradation Cause Analysis

- Idea: express causes as structural patterns + metrics

<table>
<thead>
<tr>
<th>Violation Cause</th>
<th>Architecturally Misplaced Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>A method $f$ refers to a class/interface $D$, accesses one of its fields, or calls one of its methods, causing a violation, but $f$ neither belongs to its containing class $C$ nor to the surrounding module $M$.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symptom</th>
<th>typerref</th>
<th>access</th>
<th>call</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mod SIM</td>
<td>modsim($f$) = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mod SIM</td>
<td>cohesion($f$, $C$) = 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mod SIM</td>
<td>modsim($C$) = 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Challenges
  - Interpretation of metric values for violations instances – symptoms express “ideal” case
  - Overlap with other violation causes

- Use of ML techniques
  - Classification of “safe” violation cause instances
  - Discovery of novel violation causes

ML in Recommending Repair Actions

- Refactoring recommender

- Fixed, predefined
  - Set of rules
  - Priorities of rules
  - Action parts

- Machine Learning could support
  - Adapt priorities by considering additional features and observing acceptance/rejection of recommendations
  - Observe additional manual actions after accepting recommendations and refine action parts / learn new rules

Conclusion

• We believe there is huge potential for ML in software architecture maintenance and evolution

• Access to data is expected to be a huge challenge

• Validated intended architectures and degradation analysis by experts needed for training
Thanks for your attention!

Any questions?

sebastian.herold@kau.se