Mining Software Repositories
Master Course

University of Koblenz-Landau
Faculty of Computer Science
Software Languages Team
Prof. Dr. Ralf Lämmel

Acknowledgement: Thomas Bernau has kindly helped in putting together these slides. Thank you, Thomas! :-}
Mining Software Repositories

„The Mining Software Repositories (MSR) field analyzes the rich data available in software repositories to uncover interesting and actionable information about software systems and projects.”

[MSR’15: http://2015.msrconf.org/]
Mining Software Repositories

“Software repositories such as source control systems, archived communications between project personnel, and defect tracking systems are used to help manage the progress of software projects.”

[MSR’15: http://2015.msrconf.org/]
Why?

• Support maintenance of software systems
• Improve software design/reuse
• Empirically validate novel ideas & techniques
• Understand software development & evolution
• Plan future development

[MSR’15: http://2015.msrconf.org/]
An Example

• Can we predict co-change candidates for a particular clone fragment using evolutionary coupling?
  – Inconsistently changing coupled fragments tends to introduce bugs.

• How?

Prediction and ranking of co-change candidates for clones
http://dl.acm.org/citation.cfm?doid=2597073.2597104
Prediction and ranking of co-change candidates for clones

Code clones are identical or similar code fragments scattered in a code-base. A group of code fragments that are similar to one another form a clone group. Clones in a particular group often need to be changed together (i.e., co-changed) consistently. However, all clones in a group might not require consistent changes, because some clone fragments might evolve independently. Thus, while changing a particular clone fragment, it is important for a programmer to know which other clone fragments in the same group should be consistently co-changed with that particular clone fragment.

In this research work, we empirically investigate whether we can automatically predict and rank these other clone fragments (i.e., the co-change candidates) from a clone group while making changes to a particular clone fragment in this group. For prediction and ranking we automatically retrieve and infer evolutionary coupling among clones by mining the past clone evolution history. Our experimental result on six subject systems written in two different programming languages (C, and Java) considering both exact and near-miss clones implies that we can automatically predict and rank co-change candidates for clones by analyzing evolutionary coupling. Our ranking mechanism can help programmers pinpoint the likely co-change candidates while changing a particular clone fragment and thus, can help us to better manage software clones.
Prediction and ranking of co-change candidates for clones

• How?
  – Analyze commit history and source code:
    • Identify clone fragments of the same clone class
    • Identify co-changes

http://dl.acm.org/citation.cfm?doid=2597073.2597104
Prediction and ranking of co-change candidates for clones

http://dl.acm.org/citation.cfm?doid=2597073.2597104
Prediction and ranking of co-change candidates for clones

• How?
  – Analyze commit history:
    • Identify clone fragments of the same clone class
    • Identify co-changes
  – Make prediction:

http://dl.acm.org/citation.cfm?doid=2597073.2597104
Prediction and ranking of co-change candidates for clones

http://dl.acm.org/citation.cfm?doid=2597073.2597104
So ...
• Can we predict co-change candidates for a particular clone fragment using evolutionary coupling?
  – Yes!
  – Precision: 85.18%
  – Recall: 43.17%

The predicted candidate do indeed co-change.
The co-changed candidates are indeed predicted.

http://dl.acm.org/citation.cfm?doid=2597073.2597104
Another Example

• Which are the most energy-greedy Android API methods?

• Which sequences of Android API calls are the most energy-greedy?
  – Batteries are small enough as is!

• How?
  – Trace Execution & Power Consumption
  – Line them up!

Mining energy-greedy API usage patterns in Android apps: an empirical study
http://dl.acm.org/citation.cfm?doid=2597073.2597085
Energy consumption of mobile applications is nowadays a hot topic, given the widespread use of mobile devices. The high demand for features and improved user experience, given the available powerful hardware, tend to increase the apps’ energy consumption. However, excessive energy consumption in mobile apps could also be a consequence of energy greedy hardware, bad programming practices, or particular API usage patterns. We present the largest to date quantitative and qualitative empirical investigation into the categories of API calls and usage patterns that—in the context of the Android development framework—exhibit particularly high energy consumption profiles. By using a hardware power monitor, we measure energy consumption of method calls when executing typical usage scenarios in 55 mobile apps from different domains. Based on the collected data, we mine and analyze energy-greedy APIs and usage patterns. We zoom in and discuss the cases where either the anomalous energy consumption is unavoidable or where it is due to suboptimal usage or choice of APIs. Finally, we synthesize our findings into actionable knowledge and recipes for developers on how to reduce energy consumption while using certain categories of Android APIs and patterns.
Mining energy-greedy API usage patterns in Android apps

http://dl.acm.org/citation.cfm?doid=2597073.2597085
Mining energy-greedy API usage patterns in Android apps

http://dl.acm.org/citation.cfm?doid=2597073.2597085
Mining energy-greedy API usage patterns in Android apps

So …

• Which are the most energy greedy Android API methods?
• Which sequences of Android API calls are the most energy-greedy?
• Concluding:
  – DBMS/SQL persistence is expensive
  – MVC view refreshs are expensive
  – Widget updates are expensive
  – Information hiding is expensive

http://dl.acm.org/citation.cfm?doid=2597073.2597085
Yet Another Example

• According to developers, what are the main causes for software energy consumption?
• What solutions do developers employ or recommend to save energy?
  – Same problem, different perspective!
• How?

Mining questions about software energy consumption
http://dl.acm.org/citation.cfm?doid=2597073.2597110
A growing number of software solutions have been proposed to address application-level energy consumption problems in the last few years. However, little is known about how much software developers are concerned about energy consumption, what aspects of energy consumption they consider important, and what solutions they have in mind for improving energy efficiency. In this paper we present the first empirical study on understanding the views of application programmers on software energy consumption problems. Using StackOverflow as our primary data source, we analyze a carefully curated sample of more than 300 questions and 550 answers from more than 800 users. With this data, we observed a number of interesting findings. Our study shows that practitioners are aware of the energy consumption problems: the questions they ask are not only diverse -- we found 5 main themes of questions -- but also often more interesting and challenging when compared to the control question set. Even though energy consumption-related questions are popular when considering a number of different popularity measures, the same cannot be said about the quality of their answers. In addition, we observed that some of these answers are often flawed or vague. We contrast the advice provided by these answers with the state-of-the-art research on energy consumption. Our summary of software energy consumption problems may help researchers focus on what matters the most to software developers and end users.
Mining questions about software energy consumption

• How?
  – Mine communities (StackOverflow)

http://dl.acm.org/citation.cfm?doid=2597073.2597110
Mining questions about software energy consumption

How to create a simple line graph in VB.NET for a website

Has anyone got any suggestions on how I can create a very simple graph for a web application? It is to show energy consumption levels per month for a year at a time.

So basically I want it to look like (assuming my ASCII works)

```
-- \--
```

Yeah it doesn't so I've changed my open id image to the sample

http://mrbrutal.myopenid.com/

Sorry for the long winded approach - a question for another time - free image hosting by openid (and another site probably)

Cheers

Mining questions about software energy consumption

• How?
  – **Mine communities (StackOverflow)**
  – **Use thematic analysis (e.g., LDA or Bayes Classifier)** to find common themes in questions&answers.

http://dl.acm.org/citation.cfm?doid=2597073.2597110
http://zinkov.com/images/lda_plate.png
Mining questions about software energy consumption

• How?
  – Mine communities (StackOverflow)
  – Use thematic analysis (e.g., LDA or Bayes Classifier) to find common themes in questions&answers
  – Interpret themes

http://dl.acm.org/citation.cfm?doid=2597073.2597110
Mining questions about software energy consumption

So ...

• According to developers, what are the main causes for software energy consumption?
  – Faulty GPS behaviour
  – Background activities
  – Excessive synchronization
  – Background wallpapers
  – Advertisement
  – High GPU usage

http://dl.acm.org/citation.cfm?doid=2597073.2597110
Mining questions about software energy consumption

So ...

- What solutions do developers employ or recommend to save energy?
  - Keep IO to a minimum
  - Bulk operations
  - Avoid Polling
  - Hardware Coordination
  - Concurrent programming
  - Lazy Initialization
  - Efficient Data structure

http://dl.acm.org/citation.cfm?doid=2597073.2597110
And Yet Another Example

• Can we identify and rank duplicate bug reports?
  – Otherwise, developers have to consider the duplicates and spend time on discovering the duplicate status.

• How?

Search-based duplicate defect detection: an industrial experience
http://dl.acm.org/citation.cfm?id=2487122
Duplicate defects put extra overheads on software organizations, as the cost and effort of managing duplicate defects are mainly redundant. Due to the use of natural language and various ways to describe a defect, it is usually hard to investigate duplicate defects automatically. This problem is more severe in large software organizations with huge defect repositories and massive number of defect reporters. Ideally, an efficient tool should prevent duplicate reports from reaching developers by automatically detecting and/or filtering duplicates. It also should be able to offer defect triagers a list of top-N similar bug reports and allow them to compare the similarity of incoming bug reports with the suggested duplicates. This demand has motivated us to design and develop a search-based duplicate bug detection framework at BlackBerry. The approach follows a generalized process model to evaluate and tune the performance of the system in a systematic way. We have applied the framework on software projects at BlackBerry, in addition to the Mozilla defect repository. The experimental results exhibit the performance of the developed framework and highlight the high impact of parameter tuning on its performance.
Search-based duplicate defect detection

• How?
  – Mine the bug repository (e.g., Bugzilla or Jira)
  – Use information retrieval to search for similar reports and rank them

http://dl.acm.org/citation.cfm?id=2487122
Search-based duplicate defect detection

Fig. 1. The Duplicate Defect Detection (DDD) Framework

http://dl.acm.org/citation.cfm?id=2487122
Search-based duplicate defect detection

So ...

• Can we identify and rank duplicate bug reports?
  – Yes!

http://dl.acm.org/citation.cfm?id=2487122
Search-based duplicate defect detection

![Graph showing recall rates with alternate IR algorithms](http://dl.acm.org/citation.cfm?id=2487122)

Fig. 6. Recall Rates with Alternate IR Algorithms
One more Example

• How do the QR (query reformulation) techniques that are currently in the literature perform for source code retrieval?
  – If you search for something in source code you’re not familiar with, your query may require reformulation to return good results.

• How?

Assisting code search with automatic query reformulation for bug localization http://dl.acm.org/citation.cfm?id=2487145
Source code retrieval plays an important role in many software engineering tasks. However, designing a query that can accurately retrieve the relevant software artifacts can be challenging for developers as it requires a certain level of knowledge and experience regarding the code base. This paper demonstrates how the difficulty of designing a proper query can be alleviated through automatic Query Reformulation (QR), an under-the-hood operation for reformulating a user's query with no additional input from the user. The proposed QR framework works by enriching a user's search query with certain specific additional terms drawn from the highest-ranked artifacts retrieved in response to the initial query. The important point here is that these additional terms injected into a query are those that are deemed to be "close" to the original query terms in the source code on the basis of positional proximity. This similarity metric is based on the notion that terms that deal with the same concepts in source code are usually proximal to one another. We demonstrate the superiority of our QR framework in relation to the QR frameworks well-known in the natural language document retrieval by showing significant improvements in bug localization performance for two large software projects using more than 4,000 queries.
Assisting code search for bug localization

• How?
  – Use information retrieval (e.g., vector space model) to adjust a query to the data it’s executed on.
  – Use the results for the initial query in a reformulated query.

http://dl.acm.org/citation.cfm?id=2487145
Assisting code search for bug localization

Fig. 2: An illustration of data flow in QR process.

http://dl.acm.org/citation.cfm?id=2487145
Assisting code search for bug localization

So ...

• How do the QR (query reformulation) techniques that are currently in the literature perform for source code retrieval?

http://dl.acm.org/citation.cfm?id=2487145
Assisting code search for bug localization

**TABLE III: Retrieval accuracy with QR for the three QR methods**

<table>
<thead>
<tr>
<th>QR Method</th>
<th>Eclipse</th>
<th>Chrome</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
<td>P@1</td>
<td>P@5</td>
</tr>
<tr>
<td>SCP</td>
<td>0.2296</td>
<td>0.1893</td>
<td>0.1011</td>
</tr>
<tr>
<td>RM</td>
<td>0.2154</td>
<td>0.1670</td>
<td>0.0970</td>
</tr>
<tr>
<td>ROCC</td>
<td>0.2163</td>
<td>0.1713</td>
<td>0.0961</td>
</tr>
<tr>
<td>Baseline</td>
<td>0.2089</td>
<td>0.1653</td>
<td>0.0921</td>
</tr>
</tbody>
</table>

Significant accuracy improvements compared to baseline

http://dl.acm.org/citation.cfm?id=2487145
Looking forward a great research-oriented course where you read, think, and analyze à la MSR.