On the Effect of Program Exploration on Maintenance Tasks

Zéphyrin Soh, Foutse Khomh, Yann-Gaël Guéhéneuc, Giuliano Antoniol, Bram Adams

Department of Computer and Software Engineering
École Polytechnique de Montréal, Québec, Canada

October 14, 2013
Outline

Introduction

User Study

Empirical Study

Conclusion and Future Work

Introduction

User Study

Empirical Study

Conclusion and Future Work
Program Exploration

Zéphyrin Soh et al.

Introduction

Context and Example (1/3)

Program Exploration

- Developers interact with program entities, e.g., open files, edit methods, etc.
- Developers explore the program, i.e., move from program entities to program entities.
- Types of program entities.
Containment principle: Aggregate the exploration graph at class level
Introduction

Context and Example (2/3)

Exploration graph aggregated at class level

Program Exploration
Zéphyrin Soh et al.
A graph represents a program exploration

(a) Graph 1
(b) Graph 2

- Is there any difference between exploration graphs?
- How to find the difference?
- How does the difference affect the maintenance tasks?
Introduction
Why do we care? (1/3)

Research on Program Exploration

► Developers spend on average 35% of their time navigating within and between source files [1]
► Developers perform on average 19.31 navigation actions between two modifications [2]

Introduction
Why do we care? (2/3)

Reasons of Program Exploration

- Developers may have several different ways to explore a program e.g.,
  - Looking for relevant entities ⇒ Some developers (may) not know where to look at.
  - Validate previous changes ⇒ Developers (may) know the target entity to look at.

⇒ In some cases, developers may perform more back and forth navigations (i.e., revisitation) on program entities.
Introduction

Why do we care? (3/3)

Revisitation-based Exploration

- Some recommendation systems (e.g., Mylyn [3], NavClus [4]) are based on the revisitation of program entities
- We wonder if revisitation is “good“ or “bad“
  ⇒ Knowing that is helpful to improve revisitation-based approach for recommendation systems.

In the revisitation-based exploration, we focus on two extreme cases:

- **Referenced Exploration (RE):** A developer revisits one (or a set of) entity(ies) already visited ⇒ referenced entities
- **Unreferenced Exploration (UE):** There is no set of referenced entities ⇒ Almost the same revisitation of program entity(ies)
User Study

Research Question

Exploration Strategies

- In the revisitation-based exploration, we focus on two extreme cases:
  - Referenced Exploration (RE): A developer revisits one (or a set of) entity(ies) already visited ⇒ referenced entities
  - Unreferenced Exploration (UE): There is no set of referenced entities ⇒ Almost the same revisitation of program entity(ies)

Do developers follow a referenced exploration when performing maintenance tasks?
User Study

Approach (1/5)

Data Collection

Mylyn’s IHs appear as attachment (XML file with name “mylyn-context.zip”) to a bug
User Study

Approach (1/5)

Data Collection

Mylyn’s IHs appear as attachment (XML file with name “mylyn-context.zip”) to a bug

Top four Eclipse projects with more IHs (from 2,601 bugs)

<table>
<thead>
<tr>
<th></th>
<th>ECF</th>
<th>Mylyn</th>
<th>PDE</th>
<th>Platform</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td># IHs</td>
<td>26</td>
<td>1,273</td>
<td>131</td>
<td>275</td>
<td>1,705</td>
</tr>
</tbody>
</table>
How do we build an oracle?

Random sample of IHs (proportional among projects)
User Study
Approach (2/5)

How do we build an oracle?

Random sample of IHs
(proportional among projects)
How do we build an oracle?

Random sample of IHs (proportional among projects)

9 subjects
Java experience - mean: 4.16 yrs -sd: 2.80 yrs
**User Study**

**Approach (2/5)**

*How do we build an oracle?*

- **Random sample of IHs** (proportional among projects)

- **9 subjects**
  - Java experience: mean = 4.16 yrs, sd = 2.80 yrs

---

**User Study**

**Research Question**

- **RQ1**
- **RQ2**

**Approach**

- How do we build an oracle?
- **9 subjects**
- Java experience:
  - mean: 4.16 yrs
  - sd: 2.80 yrs
- Random sample of IHs (proportional among projects)
How do we **build an oracle?**

**Random sample of IHs**
(proportional among projects)

9 subjects
Java experience
- mean : 4.16 yrs
- sd : 2.80 yrs
User Study
Approach (2/5)

How do we build an oracle?

Random sample of IHs (proportional among projects)

How much the subjects agreed?
Fliess’ Kappa interater agreement coefficient

- Overall: fair agreement (0.36)
- RE: almost moderate agreement (0.38)
- UE: almost moderate agreement (0.39)
- Doubt: No agreement (-0.009)
User Study
Approach (3/5)

How do we aggregate the results?
User Study
Approach (3/5)

How do we aggregate the results?

>= 2/3 subjects

RE  • • • •  RE  UE  • • • •  UE  D

Conclusion and Future Work
Conclusion
Threats to Validity and Future Work
User Study
Approach (3/5)

How do we aggregate the results?

>= 2/3 subjects

UE
User Study
Approach (3/5)

How do we aggregate the results?

< 2/3 subjects

D

< 2/3 subjects

D

< 2/3 subjects

D
Identification Technique

Subjects count the number of nodes and the number of in/out arrows in the graphs ⇒ look at the distribution/inequality of revisits across graphs
User Study
Approach (4/5)

Identification Technique

Subjects count the number of nodes and the number of in/out arrows in the graphs
⇒ look at the distribution/inequality of revisits across graphs

Gini Inequality Index [5]

- We measure how much classes are (in)equality revisited
- Used in econometrics to measure the inequality of income among a population
- Population = classes
  Income of a class = NumRevisits
  \[ Gini(IH) = \frac{1}{2n^2\mu} \sum_{i=1}^{n} \sum_{j=1}^{n} | NumRevisit(e_{ni}) - NumRevisit(e_{nj}) | \]

[5] K. Xu, *How has the literature on Gini’s index evolved in the past 80 years?*, Technical report, Department of Economics, Dalhouse University, 2004
User Study
Approach (5/5)

Inequality Index Threshold

Define a threshold to know if classes are (in)equally revisited

- If $Gini(IH) < Threshold$
  ⇒ Classes are almost equally revisited
  ⇒ UE: Unreferenced Exploration

- If $Gini(IH) \geq Threshold$
  ⇒ Some classes are more revisited than others
  ⇒ RE: Referenced Exploration
User Study
Approach (5/5)

Inequality Index Threshold

0
Perfect equality

1
Maximum inequality
User Study

Approach (5/5)

Inequality Index Threshold

Threshold?

Perfect equality

Maximum inequality
User Study
Approach (5/5)

Inequality Index Threshold

![Diagram showing the flow of steps in the user study approach. The process starts with 'Start', followed by 'ImportProjectSetMainPage', then 'TeamWizardPage', and finally 'End'. There are decision points at steps 2 and 3, leading to 'SendMessageProvider' or 'TeamWizardPage', respectively. The flow returns to 'End' at step 6.](image-url)
User Study
Approach (5/5)

Inequality Index Threshold

Perfect equality

0.28

Maximum inequality
Inequality Index Threshold

- Threshold = 0.2 ==> RE
- Threshold = 0.3 ==> UE
Inequality Index Threshold

Use F-Measure to maximize both precision and recall.
User Study
Approach (5/5)

Inequality Index Threshold

Use F-Measure to maximize both precision and recall
User Study

Results (1/2)

Percentage RE vs. UE

- RE: 28.03% of Interaction Histories
- UE: 71.96% of Interaction Histories

⇒ Developers follow mostly the unreferenced exploration (UE) when performing a maintenance task.
User Study
Results (1/2)

Percentage RE vs. UE

- RE: 28.03% of Interaction Histories
- UE: 71.96% of Interaction Histories

⇒ Developers follow mostly the unreferenced exploration (UE) when performing a maintenance task.

- Methodical developers do not reinvestigate methods as frequently as opportunistic developers [6]
  - RE = opportunistic developers?
  - UE = methodical developers?

⇒ We need more investigations

User Study

Results (2/2)

Confounding Factors

► Architecture of the system
User Study

Results (2/2)

Confounding Factors

- Architecture of the system
  - Number of common classes (NCC): The more A and B have CC, the more they used the same part of the system

Confounding Factors:
- Architecture of the system
- Number of common classes (NCC): The more A and B have CC, the more they used the same part of the system
User Study
Results (2/2)

Confounding Factors

- Architecture of the system
  - Number of common classes (NCC): The more A and B have CC, the more they used the same part of the system
  - Compute NCC for each pair of IH.
    - NCC pairs with same ES vs. NCC pairs with different ES ⇒ No difference.
User Study
Results (2/2)

Confounding Factors

- Architecture of the system
  - Number of common classes (NCC): The more A and B have CC, the more they used the same part of the system
  - Compute NCC for each pair of IH. NCC pairs with same ES vs. NCC pairs with different ES ⇒ No difference.
- Task interruption: The interruption time seems to push developers to be concentrate on a set of (referenced) entities (RE)
User Study

Results (2/2)

Confounding Factors

- Architecture of the system
  - Number of common classes (NCC): The more A and B have CC, the more they used the same part of the system
  - Compute NCC for each pair of IH. NCC pairs with same ES vs. NCC pairs with different ES ⇒ No difference.

- Task interruption: The interruption time seems to push developers to be concentrate on a set of (referenced) entities (RE)

- Type of the task: Bug severity because developers may have more back and forth navigation when fixing severe bugs wrt. less severe bugs
  Inequality Indexes are not significantly different for different types of bugs.
Empirical Study

Research Questions

Research Questions

Does any difference exist in maintenance time between RE and UE?

Does any difference exist in edit/navigation ratio between RE and UE?
Empirical Study

RQ1

Approach

- Overall Time spend = \[ \sum_{entity \in IH} \frac{Duration(entity)}{\#entity} \]

- Unpaired Wilcoxon test
Empirical Study

RQ1

Approach

- Overall Time spend = \( \sum_{\text{entity} \in \text{IH}} \frac{\text{Duration}(\text{entity})}{\#\text{entity}} \)
- Unpaired Wilcoxon test

Results

- The UE is on average 12.30% less time consuming than the RE.
Empirical Study

RQ2

Approach

- Edit/navigation ratio = \( \frac{\text{NumEdit}(IH)}{\text{NumEvent}(IH)} \)
- Unpaired Wilcoxon test
Empirical Study

RQ2

Approach

- Edit/navigation ratio = \( \frac{\text{NumEdit}(IH)}{\text{NumEvent}(IH)} \)
- Unpaired Wilcoxon test

Results

- An UE requires less edit/navigation ratio than a RE.
Empirical Study

Discussions

Is there a good strategy?

- **RE (more)**  
  - **UE (less)**
  - ⇒ The back and forth actions on referenced entities are time consuming

- **RE (more)**  
  - **UE (less)**  
  - ⇒ More edit actions compare to navigation actions. **Wrong edits/modif. then come back to revert/cancel?**
Empirical Study

Discussions

Is there a good strategy?

RE (more) ⇒ The back and forth actions on referenced entities are time consuming
UE (less)

⇒ More edit actions compare to navigation actions. Wrong edits/modif. then come back to revert/cancel?

RE

UE

4/8 > 3/6 (RE better)
Empirical Study

Discussions

Is there a good strategy?

RE (more) ⇒ The back and forth actions on referenced entities are time consuming
UE (less)

RE (more) ⇒ More edit actions compare to navigation actions. Wrong edits/modif. then come back to revert/cancel?
UE (less)

4/8 > 3/6 (RE better) ➞ 2/8 < 2/6 (UE better)
Conclusion and Future Work

Conclusion

Use F-Measure to maximize both precision and recall
Conclusion and Future Work

Conclusion

Use F-Measure to maximize both precision and recall
Use F-Measure to maximize both precision and recall

\[ \text{p-value} = 0.12 \]
Conclusion and Future Work

Threats to Validity and Future Work

➤ **Construct validity**: An IH can be a part of developers’ work
Conclusion and Future Work

Threats to Validity and Future Work

- **Construct validity**: An IH can be a part of developers’ work

- **Conclusion validity**: The time recorded can be different to the “real” time spent
  ⇒ Perform an experiment to collect a data
Conclusion and Future Work

Threats to Validity and Future Work

- **Construct validity**: An IH can be a part of developers’ work
- **Conclusion validity**: The time recorded can be different to the “real” time spent
  ⇒ Perform an experiment to collect a data
- **Internal validity**: We use only Mylyn’s IH
  ⇒ Use the IH from other tools
Conclusion and Future Work

Threats to Validity and Future Work

- **Construct validity**: An IH can be a part of developers’ work
- **Conclusion validity**: The time recorded can be different to the “real” time spent
  ⇒ Perform an experiment to collect a data
- **Internal validity**: We use only Mylyn’s IH
  ⇒ Use the IH from other tools
- **External validity**: Our subject projects are Eclipse-based projects
  ⇒ Use other systems
Thanks for your attention!