An Empirical Study of Supplementary Bug Fixes

Jihun Park¹, Miryung Kim², Baishakhi Ray², Doo-Hwan Bae¹

1. Korea Advanced Institute of Science and Technology
2. The University of Texas at Austin
Motivation

• Human is five times more accurate at locating errors of commission than *errors of omission* [Fry and Weimer ’10]

• Several tools recommend *supplementary changes* to reduce omission errors.

• However, there has not been a *comprehensive study of the characteristics of omission errors*. 
Study Findings

• A considerable portion (22%~33%) of bugs requires supplementary patches.

• Incomplete patches are larger in size and more scattered than regular patches.

• Predicting a supplementary fix location using clone analysis alone is insufficient.

• About 15% of supplementary change locations are beyond the scope of the direct neighbors of initial change locations.
Outline

• Research Questions
• Bug Categorization
• Study Results
• Related work
• Discussions
Research Questions

Q1. What is the extent and characteristics of supplementary changes?
Q2. What are the common causes of incomplete bug fixes?
Q3. Are supplementary bug fixes similar to corresponding initial fixes?
Q4. Where is the location of supplementary bug fixes in relation to initial fixes?
The bug IDs that were mentioned only one commit.

The bug IDs that were mentioned in multiple fix revisions.

Type 1 bug

Bug reports

Bug 22

...  

Fix commits

Fix #22

Type 2 bug

Bug 31

...  

Fix #31

Fix #31

Fix #31

Development history
The bug IDs that were mentioned only one commit.

The bug IDs that were mentioned in multiple fix revisions.

Type 1 bug

Bug reports

Bug 22

Type 2 bug

Supplementary patches

Fix commits

Fix #22

Fix #31

An initial patch

Development history

Fix #31 Fix #31 Fix #31
Q1-1. What is the extent of Type 2 bugs?

<table>
<thead>
<tr>
<th></th>
<th>Eclipse JDT core</th>
<th>Eclipse SWT</th>
<th>Mozilla project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total revisions</td>
<td>17000 revisions</td>
<td>21530 revisions</td>
<td>2000000 revisions</td>
</tr>
<tr>
<td># of bugs</td>
<td>1812</td>
<td>1256</td>
<td>11254</td>
</tr>
<tr>
<td>Type 1 bugs</td>
<td>1405 (77.54%)</td>
<td>954 (75.96%)</td>
<td>7562 (67.19%)</td>
</tr>
<tr>
<td>Type 2 bugs</td>
<td>407 (22.46%)</td>
<td>302 (24.04%)</td>
<td>3692 (32.81%)</td>
</tr>
</tbody>
</table>

22% ~ 33% bugs require supplementary bug fixes.
Q1-2. What are the characteristics of bugs that were fixed more than once?

• We examine the **time taken to fix bugs**: the time gap between “REPORTED” to “FIXED” or “CLOSED” statuses.

• We compare **the total number of developers** involved in Type 1 vs. Type 2 bugs.
Q1-2. What are the characteristics of bugs that were fixed more than once?

### The time taken to resolve bugs

<table>
<thead>
<tr>
<th></th>
<th>Type 1 bug</th>
<th>Type 2 bug</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eclipse JDT core</td>
<td>120.79</td>
<td>188.27</td>
<td>3.84e-04</td>
</tr>
<tr>
<td>Eclipse SWT</td>
<td>176.99</td>
<td>337.32</td>
<td>2.65e-07</td>
</tr>
<tr>
<td>Mozilla</td>
<td>594.50</td>
<td>805.92</td>
<td>8.40e-42</td>
</tr>
</tbody>
</table>

### The number of developers involved

<table>
<thead>
<tr>
<th></th>
<th>Type 1 bug</th>
<th>Type 2 bug</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eclipse JDT core</td>
<td>3.67</td>
<td>4.44</td>
<td>1.45e-12</td>
</tr>
<tr>
<td>Eclipse SWT</td>
<td>3.13</td>
<td>4.29</td>
<td>1.39e-09</td>
</tr>
<tr>
<td>Mozilla</td>
<td>4.70</td>
<td>7.28</td>
<td>2.05e-84</td>
</tr>
</tbody>
</table>

**More developers are involved Type 2 bugs, and they take longer to be resolved.**
Q2. What are the characteristics of incomplete patches?

- We compare patches of Type 1 bugs and initial patches of Type 2 bugs in terms of **number of files**, **patch size**, **entropy**, etc.

<table>
<thead>
<tr>
<th></th>
<th>Files</th>
<th>LOC</th>
<th>Added LOC</th>
<th>Entropy (file)</th>
<th>Entropy (package)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Type 1</td>
<td>Type 2</td>
<td>Type 1</td>
<td>Type 2</td>
<td>Type 1</td>
</tr>
<tr>
<td>Total</td>
<td>3.30</td>
<td>5.72</td>
<td>147.98</td>
<td>309.38</td>
<td>60.92</td>
</tr>
<tr>
<td>p-value</td>
<td>1.15E-18</td>
<td>4.46E-05</td>
<td>4.03E-12</td>
<td>2.05E-09</td>
<td>1.04E-10</td>
</tr>
</tbody>
</table>

**Initial patches of Type 2 bugs are larger in size, and more scattered than patches of Type 1 bugs.**
Q2. What are the causes of incomplete bug fixes?

- **Manual investigation** on 100 supplementary patches

<table>
<thead>
<tr>
<th>Types</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. An initial patch is ported to a different component or branch.</td>
<td>28%</td>
</tr>
<tr>
<td>2. The <em>conditional statement</em> of an initial fix is not correct.</td>
<td>23%</td>
</tr>
<tr>
<td>3. Code elements <em>referring to or being referenced</em> by changed code are later updated.</td>
<td>15%</td>
</tr>
<tr>
<td>4. Two different parts <em>calling different subclasses of the same type</em> are not updated together.</td>
<td>4%</td>
</tr>
<tr>
<td>5. <strong>Incomplete refactoring</strong> induces a supplementary patch</td>
<td>3%</td>
</tr>
</tbody>
</table>

The common causes of incomplete fixes include missed porting updates, incorrect conditional statements, and incomplete refactoring.

* more types are on the paper
Q3. Are supplementary bug fixes similar to corresponding initial fixes?

- Study method: We identify similar patches using clone analysis tool (CCFinder, Kamiya et al. 2002)

<table>
<thead>
<tr>
<th>Software</th>
<th>Cloned Patches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eclipse JDT core</td>
<td>424 (70.20%)</td>
</tr>
<tr>
<td>Eclipse SWT</td>
<td>392 (68.54%)</td>
</tr>
<tr>
<td>Mozilla</td>
<td>5477 (78.75%)</td>
</tr>
</tbody>
</table>
Q3. Are supplementary bug fixes similar to corresponding initial fixes?

- Study method: We identify similar patches using clone analysis tool (CCFinder, Kamiya et al. 2002)
- We exclude backported patches, because they are simply identical patches applied to different branch locations.

![Diagram showing the percentage of cloned and backported patches in Eclipse JDT core, Eclipse SWT, and Mozilla.](attachment:diagram.png)
Q3. Are supplementary bug fixes similar to corresponding initial fixes?

• Study method: We identify similar patches using clone analysis tool (CCFinder, Kamiya et al. 2002)

• We exclude backported patches, because they are simply identical patches applied to different branch locations.

• Only 12%, 25%, and 9% include at least five similar lines.

Predicting a supplementary fix location using code clone analysis alone is insufficient.
Q4. Where is the location of supplementary bug fixes in relation to initial fixes?

- 48% and 42% are made at the similar line location of an initial patch. (similar heuristics with Yin et al. ‘11)
Q4. Where is the location of supplementary bug fixes in relation to initial fixes?

- **48% and 42%** are made at the similar line location of an initial patch. (similar heuristics with Yin et al. ‘11)
- **32% and 29%** have **structural dependences** with an initial patch. (using LSdiff by Kim and Notkin ‘09)
Q4. Where is the location of supplementary bug fixes in relation to initial fixes?

- **48% and 42%** are made at the similar line location of an initial patch. (similar heuristics with Yin et al. ‘11)
- **32% and 29%** have **structural dependences** with an initial patch. (using LSdiff by Kim and Notkin ‘09)

About **15%** of supplementary change locations are beyond the scope of the direct neighbors of initial patch locations.
Related Work

• 17% to 45% of fixes are recurring and they can be identified using similar code units. (Nguyen et al.)

• 14.8% to 24.4% of post release patches are incorrect. (Yin et al.)

• 9% of all bugs are re-opened. (Gu et al.)
Limitations and Future Work

• Expand the studied period of bug reports.
• Investigate the relationship among supplementary fixes.
• Develop new tools for reducing incomplete bug fixes. (e.g., detection of incomplete refactoring)
Summary

• A considerable portion of bugs requires supplementary patches.

• Incomplete patches are larger in size and more scattered than regular patches.

• Predicting a supplementary fix location using code clone analysis alone is insufficient.

• About 15% of supplementary change locations are beyond the scope of the direct neighbors of initial patch locations.
Acknowledgements

• This research was sponsored by the Agency for Defense Development, Republic of Korea, under the grant UD100031CD.

• This work was in part supported by National Science Foundation under the grants CCF-1149391, CCF-1043810, and CCF-1117902, and by a Microsoft SEIF award.
Thank You!

Jihun Park
Email: jhpark@se.kaist.ac.kr
PhD Candidate
Software Engineering LAB
Computer Science Dept.
KAIST, Daejeon, South Korea