The Effects of Enhanced Compiler Error Messages on a Syntax Error Debugging Test

BRETT A BECKER*
KYLE GOSLIN, GRAHAM GLANVILLE

UNIVERSITY COLLEGE DUBLIN
CCT COLLEGE DUBLIN

*BRETT DOT BECKER AT UCD DOT IE
The Problem With Compiler Error Messages

• One of the many challenges novice programmers face from the time they write their first program is inadequate compiler error messages.

• These messages are:
  • Immediate
  • Often the only feedback students get
  • Essential for correcting code (of course)
  • Come from a machine that students think is infallible

• But they are very often…
The (50+ year old) Problem With Compiler Error Messages

inadequate and not understandable (Moulton and Muller, 1967), useless (Wexelblat, 1976), not optimal (Litecky and Davis, 1976), inadequate (again, 17 years later) (Brown, 1983), frustrating (Flowers et al., 2004), cryptic and confusing (Jadud, 2006), notoriously obscure and terse (Ben-Ari, 2007), undecipherable (Traver, 2010), still very obviously less helpful than they could be (McCall and Kölling, 2014), inscrutable (Ko, 2017), and a source of frustration and a barrier to progress (Becker et al., 2016).

See Becker et al. (2016) for more, and for references.

Recent Results on Enhancing Compiler Error Messages

• Increased activity since ~2010

• Some seemingly conflicting results

• Different studies have attempted to measure the effects of enhanced compiler error messages on different metrics:
  • Error frequency, overall number of errors, repeated errors, errors per student
  • Attempts required to resolve errors
  • Non-compiling submissions
  • Student progress, test scores, academic performance
  • Time to resolve errors
Approach

- Most prior studies measure how many errors students produce, and then possibly rectify while writing programs (or some derived metric based on this).
- We measured how many pre-existing syntax errors are rectified by students while debugging programs.
- Specifically, we measured the effect of enhanced compiler error messages in an empirical control/intervention experiment where students were given the task of removing syntax errors from non-compiling source code they did not write.
Methodology

• 86 CS1 (BSc) students studying Java in separate groups

• Utilized custom Java editor that can enhance error messages
  • Half of students received standard javac error messages
  • Other half received enhanced error messages
  • Both groups took the same debugging test under the same conditions
The Test

• ~100 (fairly sparse) lines of code, available at www.brett.becker.com/sigcse18/

• 24 occurrences of 11 different syntax errors (some appeared more than once)
  • Distribution intended to mimic actual distribution (some common and frequent errors, some more rare and infrequent errors)

• Test divided into 5 logical tasks with known, pre-specified output (once errors are rectified)
  • This attempted to mitigate for illegitimate fixes (e.g. deleting line of code with error)
The Test

<table>
<thead>
<tr>
<th>Compiler error message type</th>
<th>Quantity</th>
<th>Rank</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>cannot find symbol</td>
<td>4</td>
<td>1</td>
<td>19%</td>
</tr>
<tr>
<td>'.' expected</td>
<td>4</td>
<td>3</td>
<td>12%</td>
</tr>
<tr>
<td>not a statement</td>
<td>4</td>
<td>4</td>
<td>9%</td>
</tr>
<tr>
<td>illegal start of expression</td>
<td>3</td>
<td>5</td>
<td>7%</td>
</tr>
<tr>
<td>bad operand types for binary operator $op$</td>
<td>1</td>
<td>8</td>
<td>5%</td>
</tr>
<tr>
<td>incompatible types</td>
<td>2</td>
<td>9</td>
<td>4%</td>
</tr>
<tr>
<td>&lt;identifier&gt; expected</td>
<td>1</td>
<td>10</td>
<td>4%</td>
</tr>
<tr>
<td>variable $v_name$ is already defined in method $m_name$</td>
<td>2</td>
<td>13</td>
<td>3%</td>
</tr>
<tr>
<td><code>)</code> expected</td>
<td>1</td>
<td>16</td>
<td>&lt;2%</td>
</tr>
<tr>
<td>missing return statement</td>
<td>1</td>
<td>17</td>
<td>&lt;2%</td>
</tr>
<tr>
<td>package $p_name$ does not exist</td>
<td>1</td>
<td>25</td>
<td>&lt;1%</td>
</tr>
</tbody>
</table>

- **Compiler errors included in test**
  - **Quantity** is the number of errors in each test generating the corresponding compiler error message type
  - **Rank** is the overall frequency rank from control group* (1 is most frequent)
  - **Frequency** is the frequency, from control group*

The Test

• In analyzing the submissions, we tabulated the following:

  1. whether the submission compiled

  2. if the program compiled, whether the output of each of the five tasks was correct

  3. whether each of the 24 errors were rectified in a manner consistent with the specified output

  4. a student score based on number of errors rectified
Research Questions

- RQ1: Do enhanced compiler error messages have an effect on the number of successfully compiling submissions?
- RQ2: Do enhanced compiler error messages have an effect on the number of tasks completed correctly?
- RQ3: Do enhanced compiler error messages have an effect on the number of rectified syntax errors?
- RQ4: Do enhanced compiler error messages have an effect on student scores?
Results

• RQ1: Do enhanced compiler error messages have an effect on the number of *successfully compiling submissions*?

• The intervention group submitted 20 compiling submissions and the control group submitted 17. A chi-squared test indicated that the difference between groups is not significant

\[ \chi^2 = 0.43, \ p = 0.513 \ (\alpha = 0.05) \]
Results

- RQ2: Do enhanced compiler error messages have an effect on the number of tasks completed correctly?

<table>
<thead>
<tr>
<th>Task</th>
<th>Control</th>
<th>Intervention</th>
<th>$\chi^2$</th>
<th>$p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>10</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>B</td>
<td>12</td>
<td>13</td>
<td>0.06</td>
<td>0.813</td>
</tr>
<tr>
<td>C</td>
<td>13</td>
<td>12</td>
<td>0.06</td>
<td>0.813</td>
</tr>
<tr>
<td>D</td>
<td>13</td>
<td>13</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>E</td>
<td>10</td>
<td>12</td>
<td>0.24</td>
<td>0.621</td>
</tr>
</tbody>
</table>

- No differences are statistically significant
Results

• RQ3: Do enhanced compiler error messages have an effect on the number of rectified syntax errors?

• The control group rectified 665 (64.4%) of their errors while the intervention group rectified 790 (76.6%). A chi-squared test indicated that this is statistically significant $\chi^2 = 36.40, p < 0.001 (\alpha = 0.05)$
Results

- RQ3: We next sought to determine which compiler error messages contributed to this result.

<table>
<thead>
<tr>
<th>Compiler Error Message Type</th>
<th>Total Errors</th>
<th>Rectified (Control)</th>
<th>Rectified (Intervention)</th>
<th>$\chi^2$</th>
<th>$p$</th>
<th>Sig (B-H)?</th>
<th>Sig *?</th>
</tr>
</thead>
<tbody>
<tr>
<td>cannot find symbol</td>
<td>172</td>
<td>121</td>
<td>145</td>
<td>9.55</td>
<td>0.003</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td><code>:</code> expected</td>
<td>172</td>
<td>111</td>
<td>131</td>
<td>5.57</td>
<td>0.018</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>not a statement</td>
<td>172</td>
<td>116</td>
<td>135</td>
<td>5.32</td>
<td>0.021</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>illegal start of expression</td>
<td>129</td>
<td>87</td>
<td>95</td>
<td>1.19</td>
<td>0.275</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>bad operand types for binary operator $op$</td>
<td>43</td>
<td>23</td>
<td>28</td>
<td>1.20</td>
<td>0.272</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>incompatible types</td>
<td>86</td>
<td>57</td>
<td>70</td>
<td>5.09</td>
<td>0.024</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>&lt; identifier &gt; expected</td>
<td>43</td>
<td>23</td>
<td>30</td>
<td>2.41</td>
<td>0.121</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>variable $v_name$ is already defined in method $m_name$</td>
<td>86</td>
<td>49</td>
<td>57</td>
<td>1.57</td>
<td>0.210</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td><code>)</code> expected</td>
<td>43</td>
<td>35</td>
<td>37</td>
<td>0.34</td>
<td>0.559</td>
<td>N</td>
<td>N/A</td>
</tr>
<tr>
<td>missing return statement</td>
<td>43</td>
<td>13</td>
<td>23</td>
<td>4.78</td>
<td>0.029</td>
<td>Y</td>
<td>N/A</td>
</tr>
<tr>
<td>package $p_name$ does not exist</td>
<td>43</td>
<td>30</td>
<td>39</td>
<td>5.94</td>
<td>0.015</td>
<td>Y</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Sig (B-H)? indicates statistical significance (or lack of) with Benjamini-Hochberg correction for multiple tests.

Results

• RQ4: Do enhanced compiler error messages have an effect on student scores?

• We calculated a score per student based on the number of errors corrected (perfect score = 24). The average/median scores were 15.16/16.5 (control) and 17.95/21 (intervention).

• A Shapiro-Wilk test revealed neither distribution was normal.

• A Mann-Whitney U test (two-tail, independent samples) showed no statistically significant difference between groups
  \[ U = 805.5, p = 0.172 \ (\alpha = 0.05) \]
Discussion

- RQ1: Do enhanced compiler error messages have an effect on the number of successfully compiling submissions? **No**
- RQ2: Do enhanced compiler error messages have an effect on the number of tasks completed correctly? **No**
- RQ3: Do enhanced compiler error messages have an effect on the number of rectified syntax errors? **Yes**
  - 6 error messages showed a statistically significant reduction in number of rectified errors, and 3 of these also showed an effect in Becker (2016)
- RQ4: Do enhanced compiler error messages have an effect on student scores? **No**
Threats to validity

- Possible that a given error could be successfully resolved in a number of ways
- It is probable that a different test, but with the same number of the same syntax error message types, would generate different results
- Evidence that fewer control students attempted the final third of test*
  - Prather et al. (2017) observed similar behavior
  - The enhanced error messages may explain this
  - This also may be a reason that an effect was observed for some error messages

Conclusions

• This study seemed to solidify one possible view on the current state of compiler error enhancement:
  • Some studies agree, others disagree
  • Some metrics that are ‘close to the data’ such as error frequency seem to be affected by enhanced compiler error messages
  • Other derived metrics that are ‘further from the data’ seem to not be affected by enhanced compiler error messages
Moving Forward

• Back to the drawing board?

  • Study the readability of compiler error messages?

  • Do we need to apply more rigor in designing enhanced compiler error messages?
Thank you!

Questions?