The Effect of Preprocessor Annotations on Code Comprehension

A Developer’s Perception

FOSD Meeting 2019
Tue. 12 – Fri. 15 March 2019, Weimar, Germany

Maria Kanyshkova, Wolfram Fenske, Sandro Schulze

Databases and Software Engineering
University of Magdeburg, Germany
C Preprocessor Annotations: Frequently Used & Often Criticized

- “#ifdef considered harmful” [Spencer & Collyer, USENIX’92; Favre, IWPC’97; Ernst et al., TSE’02; …]
- #ifdefs may be associated with faults [e.g., Medeiros et al., GPCE’13 & ECOOP’15; GPCE’15; Ferreira et al., SPLC’16]
- #ifdefs hurt program comprehension [e.g., Walkingshaw et al., VL/HCC’11; Melo et al., ICSE’16]
- Programmers don’t like undisciplined #ifdefs [e.g., Medeiros et al., TSE’17; Malaquias et al., ICPC’17]
Variability-Aware Code Smells

- Inappropriate use of #ifdefs may be a code smell [Fenske & Schulze, VaMoS’15]
- Variability-aware code smells can be detected with metrics [Fenske et al., SCAM’15]
- More #ifdef use doesn’t affect change-proneness (much) [Fenske et al., GPCE’17]
Metrics of #ifdef Use

Do metrics of #ifdef use reflect how developers perceive the code?
Methodology

- Online questionnaire
- Five heavily annotated C functions
- 1/2 of participants received refactored functions with simplified annotations
Methodology
Original vs. Refactored Function
Methodology

Tasks

- Questionnaire sent to ~6000 GitHub developers
- Tasks for each function
  1. Two comprehension tasks
  2. Rate general code quality (e.g., regarding understandability)
  3. Specifically rate quality of #ifdef use
  4. Optionally explain why #ifdef use was inappropriate
- Received ~1000 responses
Preliminary Results
Program Comprehension Tasks

1. Which of the following statements is true? (CD02)
   - If UNIX is not defined, the function will be empty.
   - If UNIX is defined, the function will return another function.
   - Whether UNIX is defined or not has no effect on the result.
   - None of the above is true.
   - I don't know

2. When would line 20 be executed? (CD07)
   Choose a combination of conditions that would lead to the desired execution.
   - Never
   - Always
   - UNIX is defined
   - UNIX is undefined
   - BACKSLASH_IN_FILENAME is defined and the filename contains \n   - Either MSWIN or DJGPP is defined and the filename contains ~
   - The filename contains .. or //
   - The filename does not contain .. or //
   - The function vim_isAbsName(fname) returns 0
   - The function vim_isAbsName(fname) returns 1
   - I don't know

Fewer correct answers for
more #ifdefs

More correct answers for
• more nesting,
• more negation,
• longer functions,
• higher percentage of annotated lines

No consistent effect:
• number of feature constants
Preliminary Results
Ratings of Code Quality

4. Do you consider the use of preprocessor annotations in the example appropriate? [CD12]

Preprocessor annotations are directives like #define, #ifdef, and #ifndef.

- Yes
- No, because

No significant correlations between #ifdef metrics and ratings.

6. Please rate the presented code regarding the following questions: [CD32]

How easy was it to understand this code?

- Very hard
- Moderately hard
- Moderately easy
- Very easy

How easy would it be to maintain this code (e.g., to change code or fix bugs)?

How easy would it be to extend this code?

How easy would it be to detect bugs in this code?
Preliminary Results
Qualitative Assessments

Emacs

- Just ugly, makes me uncomfortable
- Please stop the torture :-)

Vim

- JESUS THAT'S A LOT OF IFDEFS
- I've written a lot of C code in my life, and maintained far more, and this is just way messy.
- I'm now considering giving up using VIM.
Preliminary Results
Were the refactorings beneficial?

<table>
<thead>
<tr>
<th>Outcome</th>
<th>Significant?</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correctness: comprehension task 1</td>
<td>not significant</td>
<td>(negative tendency)</td>
</tr>
<tr>
<td>Correctness: comprehension task 2</td>
<td>significant (p&lt;0.002)</td>
<td>negative: -7% (59% vs. 52% correct answers)</td>
</tr>
<tr>
<td>Rating of general quality (understandability, maintainability, …)</td>
<td>significant (p&lt;0.05)</td>
<td>negligible positive effect (Cliff’s delta ~0.05—0.07)</td>
</tr>
<tr>
<td>Is #ifdef use appropriate?</td>
<td>significant (p&lt;0.001)</td>
<td>positive: +12% (51% vs. 63% positive ratings)</td>
</tr>
</tbody>
</table>

Comparing refactored to original functions …
1. #ifdef use was rated more appropriate, but …
2. Performance in comprehension tasks was worse!
Conclusion & Future Work

- Large-scale questionnaire on effect of preprocessor annotations on code comprehension
- Preliminary results are inconclusive
- Possibly interesting: Actual developer performance may not align with subjective ratings.
- Future work
  - Publish data set
  - Finish data cleaning & analysis
  - Follow-up interviews to gain more insights