Black Box Software Testing

2004 Academic Edition

Part 28 -- EDITING BUGS

by

Cem Kaner, J.D., Ph.D.

Professor of Software Engineering

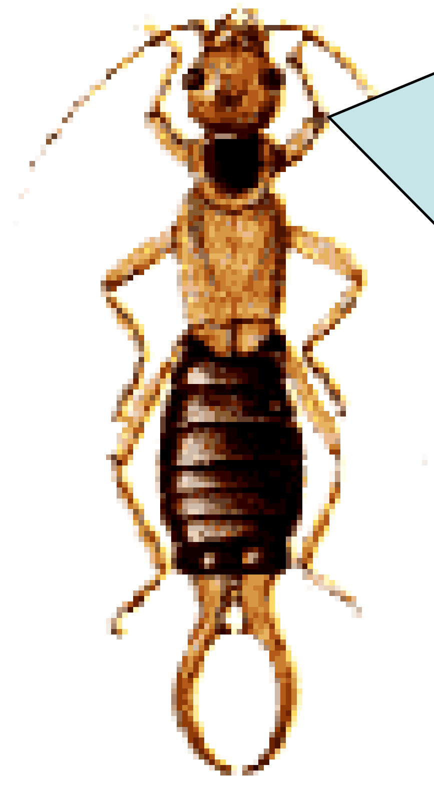
Florida Institute of Technology

and

James Bach

Principal, Satisfice Inc.

These notes are partially based on research that was supported by NSF Grant EIA-0113539 ITR/SY+PE: "Improving the Education of Software Testers." Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.



Editing bug reports

- Improve them
- Evaluate them

Editing Bug Reports

- Some groups have a second tester (usually a senior tester) review reported defects before they go to the programmer. The second tester:
  - checks that critical information is present and intelligible
  - checks whether she can reproduce the bug
  - asks whether the report might be simplified, generalized or strengthened.
- If there are problems, she takes the bug back to the original reporter.
  - If the reporter was outside the test group, she simply checks basic facts with him.
  - If the reporter was a tester, she points out problems with an objective of furthering the tester's training.

Editing Bug Reports

- This tester might review:
  - all defects
  - all defects in her area
  - all of her buddy's defects.
- In designing a system like this, beware of overburdening the reviewing testers. The reviewer will often go through a learning curve (learning about parts of the system or types of tests that she hasn't studied before). This takes time. Additionally, you have to decide whether the reviewer is doing an actual reproduction of the test or thinking about the plausibility and understandability of the report when she reads it.

Editing Bugs--Practice at Home

- Go through your bug database and find some bugs that look interesting
  - Do an initial review of them
  - Replicate them
  - Revise the descriptions to make them clearer and more useful.
- Assignment:
  - Give two improved bugs to a co-worker
  - Review two improved bugs from a co-worker
  - Compare notes
- (Note: When I teach this course to undergraduates, I require them to successfully edit bugs before they can write any. It is effective training.)

Editing Bugs Assignment Procedure

- First times: The tester gives you the bug report before entering it into the bug tracking system.
  - The reporter should give you a hard copy of the proposed bug report or a file in a format you can read. If you can't read the reporter's file format, the reporter has to give you the bug in some other format. This is the reporter's responsibility, not yours.
  - Read over the report. If you can't understand it or if there are obvious problems, note those problems and return it to the reporter. If there are significant problems when you try to read the report, don't spend any time trying to replicate it. Just give it back and deal with it again later, when it has been fixed.
  - If the report is OK to read (not perfect, but OK), make some comments (maybe on a printout, maybe in the text file that the reporter gave you) and then try to replicate the bug. Make comments as appropriate. Then hand the commented report back to the reporter. The reporter can review your comments, decide what to change, and then either:
    - Submit the bug directly into the bug tracking system, or
    - Give the bug back to you for a second review.
  - You are only obligated to review a bug once. If you review the bug and bounce it because it is unintelligible, you don't have to accept it back for replication. If you replicated it and gave feedback, you don't have to review the improved version.
  - If the reporter is submitting a bug to you that was previously reviewed by someone else, she MUST give you a copy of the report that she gave to that other person and their comments, along with the new improved report.

Editing Bugs Assignment Procedure

Later times:

- If the tester gives you the bug report before entering it into the bug tracking system,
  - Same procedure as before
- If the tester gives you the report AFTER entering it into the bug tracking system
  - Review the report for clarity and tone (see "first impressions", next slide) and send comments back to the reporter by email
  - Attempt to replicate the bug and send comments back to the reporter by email on the replication steps, your overall impressions, and any follow-up tests you recommend
  - You may edit the bug report yourself, but ONLY in the following ways.
    - Add a comment indicating that you successfully replicated the bug on XXX configuration in YYY build. (This is only valuable if the configuration or build is different from the reporter's.)
    - Add a comment describing a simpler set of replication steps. Make sure these are clear and accurate.
    - Add a comment describing why this bug would be important to customers (this is only needed if the bug looks minor or like it won't be fixed. It is only useful if you clearly know what you are talking about, your tone is respectful).
    - Your comments should NEVER appear critical or disrespectful of the original report or of the person who wrote it. You are adding information, not criticizing what was there.
  - If you edit the report in the database, never change what the reporter has actually written. You are not changing his work, you are adding comments to it at the end of the report
  - Your comments should have your name and the comment date, usually at the start of the comment, for example: "(Cem Kaner, 12/14/01) Here is an alternative set of replication steps:")
  - Send the reporter an email, telling her that you have reviewed the report and made changes.

Editing Bugs—Notes to the Bug Reporter

If you are the tester who wrote the bug report (the reporter), send me a note when you have written a report, entered it into the database and consider it finished. Here are some ground rules:

- Your first two bugs must go through an editing pass by a bug replicator before you can put them into the database. After that, it is up to you whether to submit bugs to replicators or just enter them into the database directly.
- It is up to you to make changes or not make changes to your report, based on the editor's comments. This is your bug report, it carries your signature, write it your way.
- Never, ever write a report under someone else's name.
- If you submitted a report to one editor and their edit was not helpful, you can submit it to another editor. However, when you do this, you must give the second editor a copy of the report that you gave to the first editor and a copy of the first editor's comments.
- When you are satisfied with your bug report, enter it into the bug tracking system in the form you consider final, and send me a note telling me where to find the report. I'll look it over and assign credit (or not).

Editing Bugs—First impressions

- Is the summary short (about 50-70 characters) and descriptive?
- Can you understand the report?
  - As you read the description, do you understand what the reporter did?
  - Can you envision what the program did in response?
  - Do you understand what the failure was?
- Is it obvious where to start (what state to bring the program to) to replicate the bug?
- Is it obvious what files to use (if any)? Is it obvious what you would type?
- Is the replication sequence provided as a numbered set of steps, which tell you exactly what to do and, when useful, what you will see?

Editing Bugs—First impressions

- Does the report include unnecessary information, personal opinions or anecdotes that seem out of place?
- Is the tone of the report insulting? Are any words in the report potentially insulting?
- Does the report seem too long? Too short? Does it seem to have a lot of unnecessary steps? (This is your first impression—you might be mistaken. After all, you haven't replicated it yet. But does it LOOK like there's a lot of excess in the report?)
- Does the report seem overly general ("Insert a file and you will see" – what file? What kind of file? Is there an example, like "Insert a file like blah.foo or blah2.fee"?)

Editing Bugs—Replicate the Report

- Can you replicate the bug?
- Did you need additional information or steps?
- Did you get lost or wonder whether you had done a step correctly? Would additional feedback (like, "the program will respond like this...") have helped?
- Did you have to guess about what to do next?
- Did you have to change your configuration or environment in any way that wasn't specified in the report?
- Did some steps appear unnecessary? Were they unnecessary?
- Did the description accurately describe the failure?
- Did the summary accurate describe the failure?
- Does the description include non-factual information (such as the tester's guesses about the underlying fault) and if so, does this information seem credible and useful or not?

Editing Bugs—Follow-Up Tests

- Are there follow-up tests that you would run on this report if you had the time?
  - *In follow-up testing, we vary a test that yielded a less-than-spectacular failure. We vary the operation, data, or environment, asking whether the underlying fault in the code can yield a more serious failure or a failure under a broader range of circumstances.*
  - *You will probably NOT have time to run many follow-up tests yourself. For evaluation, my question is not what the results of these tests were. Rather it is, what follow-up tests should have been run—and then, what tests were run?*
- What would you hope to learn from these tests?
- How important would these tests be?

Editing Bugs—Follow-Up Tests

- Are some tests so obviously probative that you feel a competent reporter would have run them *and described the results?*
  - The report describes a corner case without apparently having checked non-extreme values.
  - Or the report relies on other specific values, with no indication about whether the program just fails on those or on anything in the same class (what is the class?)
  - Or the report is so general that you doubt that it is accurate ("Insert any file at this point" – *really? Any file? Any type of file? Any size? Did the tester supply* reasons for you to believe this generalization is credible? Or examples of files that actually yielded the failure?)

Editing Bugs—Tester's evaluation

- Does the description include non-factual information (such as the tester’s guesses about the underlying fault) and if so, does this information seem credible and useful or not?
- Does the description include statements about why this bug would be important to the customer or to someone else?

The report need not include such information, but if it does, it should be credible, accurate, and useful.