



Software Testing 1
CSE 3411 SWE 5411

Assignment #1
Replicate and Edit Bugs

Editing Bugs

The purpose of this assignment is to give you experience editing bugs written by other people. This task will give you practice thinking about what a professional report should be, before you start entering your own reports into this public system.

- Work with the current version of Open Office Calc
- Join the QA project, use your FIT email address.
- Read the instructions at http://qa.openoffice.org/issue_handling/project_issues.html and <http://qa.openoffice.org/servlets/ProjectIssues>. Read the bug entry guidelines at http://qa.openoffice.org/issue_handling/bug_writing_guidelines.html.
- Find 5 bug reports in Issue Tracker about problems with Open Office Calc that appear to have not yet been independently verified. These are listed in the database as “unconfirmed”. If you search by project, you *may* find relevant bugs in the sc project (spreadsheet), documentation project, dba (database), api (application programming interface), and ui (user interface).
- For each report, review and replicate the bug, and add comments *as appropriate* in the Additional Comments field to the report on issuezilla.
- The assignment that you submit *to me* should list the bug numbers. I will read the comments you filed (if any) on the bug report. In addition, for each bug, tell me what was done well, what was done poorly and what was missing that should have been there in the bug report.

Editing Bugs

Assignment Procedure

For each bug:

- Review the report for clarity and tone (see “first impressions”, next slide).
 - » Include comments on clarity and tone on the memo you send me (but don’t make these comments on the bug report itself)
- Attempt to replicate the bug.
 - » Send comments to me on the replication steps (were the ones in the report clear and accurate), your overall impressions of the bug report as a procedure description, and describe any follow-up tests that you would recommend.
- You may edit the bug report yourself, primarily in the following ways.
 - » Add a comment indicating that you successfully replicated the bug on XXX configuration in YYY build.
 - » Add a comment describing a simpler set of replication steps (if you have a simpler set). 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 know what you are talking about).
 - » 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:”)

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 accurately 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 likely to yield important information 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.