How to Evaluate a Bug Report

This collection of questions helps me quickly spot (and name) the weaknesses of bug reports. Don't ask every question about every report. When writing an evaluation, highlight those answers that seemed the most insight producing.
Do ask **at least one** question within each of the four categories:
1. What are your **first impressions** of the report?
2. What happens when you attempt to **replicate the report**?
3. What **follow-up tests** should have been done in the course of writing this report?
4. Does the report include **speculation or evaluation** by the tester, and if so, is it appropriate and useful?
Skim through this list as you read the report—don't work through every question. Your evaluation should point out the strengths of what you have read as well as the weaknesses.

Evaluation: First impressions

• Is there a summary?
• Is it 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?
• 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"?)

Evaluation: Replicate the Report

• Can you replicate the bug?
• Did you need additional information or steps?
• Did you have to guess about what to do next?

• 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 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?

Evaluation: 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 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?
• 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?)

Evaluation: Tester's Speculation or Evaluation

Some bug reports include evaluative (rather than factual) remarks from the tester, such as hypotheses about the underlying fault or about the importance of the problem to the customer. The report need not include such information, but if it does, it should be credible, accurate, and useful.
• If the report includes such information, is it credible and useful or not?
• Does the report cite facts to support the evaluation?