BUG REPORTS THAT MAKE SENSE

by Mary Decker
Aldebaron Financial Solutions

Introduction

After a defect has been found, it must be reported to development so that it can be fixed. Much has been written about identifying defects and reproducing them– but very little has been done to explain the reporting process and what developers really need.

The purpose of this paper is to provide a guideline for what information should be included in a report, and how those requirements will vary based on the type of bug and the type of function.

Overview of Bugs

No matter what a system does, what language it's written in, what platform it's run on, whether it's client/server based or not– its basic functions are the same. They are broken down into the following categories:

Entry
Storage
Output
Process

As the interaction between data and the system increases usually so does the severity of the bug, and the detail needed in a report.

Bug severity can be categorized as follows:

Cosmetic
Inconvenience
Loss of Function
System crash or hang
Loss of Data

Cosmetic bugs are the simplest bugs to report, and affect the system the least. They are simply instances where things look wrong. Spelling errors, screen anomalies– these are cosmetic bugs.

Bugs that are classified as an inconvenience are just that, something that makes the system harder to use. These are slightly more nebulous since part of their effect is subjective. This also makes it harder to describe what the actual problem is.

When a bug results in a loss of function, reporting becomes a bit more complicated and the urgency to fix the bug is greater. These bugs do not affect the data, but it means that a process is useless until it is fixed. Because of this, the report again becomes more complicated.

Bugs that cause the system to crash or hang can be the hardest to reproduce, and therefore the hardest to adequately describe. If you experience a crash or hang in testing, it is imperative to see if you can reproduce the problem, documenting all the steps taken along the way. On these occasions, it is also important to include the data used in causing the system to crash/hang.

The final classification is the worst– bugs that result in the loss of data. Data is the heart of almost every system, and anything that threatens the integrity of that data must be fixed as quickly as possible. Therefore more than any other bug type it must be documented as thoroughly as possible.

Reporting Guidelines

The key to making a good report is providing the development staff with as much information as necessary to reproduce the bug. This can be broken down into 5 points:

1) Give a brief description of the problem
2) List the steps that are needed to reproduce the bug or problem
3) Supply all relevant information such as version, project and data used.
4) Supply a copy of all relevant reports and data including copies of the expected results.
5) Summarize what you think the problem is.

When you are reporting a defect the more information you supply, the easier it will be for the developers to determine the problem and fix it.

Simple problems can have a simple report, but the more complex the problem– the more information the developer is going to need.

For example: cosmetic errors may only require a brief description of the screen, how to get it and what needs to be changed.

However, an error in processing will require a more detailed description, such as:

1) The name of the process and how to get to it.
2) Documentation on what was expected. (Expected results)
3) The source of the expected results, if available. This includes spread sheets, an earlier version of the software and any formulas used)
4) Documentation on what actually happened. (Perceived results)
5) An explanation of how the results differed.
6) Identify the individual items that are wrong.
7) If specific data is involved, a copy of the data both before and after the process should be included.
8) Copies of any output should be included.

As a rule the detail of your report will increase based on a) the severity of the bug, b) the level of the processing, c) the complexity of reproducing the bug.

Anatomy of a bug report

Bug reports need to do more than just describe the bug. They have to give developers something to work with so that they can successfully reproduce the problem.

In most cases the more information– correct information– given the better. The report should explain exactly how to reproduce the problem and an explanation of exactly what the problem is.

The basic items in a report are as follows:

Version: This is very important. In most cases the product is not static, developers will have been working on it and if they've found a bug– it may already have been reported or even fixed. In either case, they need to know which version to use when testing out the bug.

Product: If you are developing more than one product– Identify the product in question.

Data: Unless you are reporting something very simple, such as a cosmetic error on a screen, you should include a dataset that exhibits the error.

If you're reporting a processing error, you should include two versions of the dataset, one before the process and one after. If the dataset from before the process is not included, developers will be forced to try and find the bug based on forensic evidence. With the data, developers can trace what is happening.

Steps: List the steps taken to recreate the bug. Include all proper menu names, don't abbreviate and don't assume anything.

After you've finished writing down the steps, follow them - make sure you've included everything you type and do to get to the problem. If there are parameters, list them. If you have to enter any data, supply the exact data entered. Go through the process again and see if there are any steps that can be removed.

When you report the steps they should be the clearest steps to recreating the bug.

Description: Explain what is wrong - Try to weed out any extraneous information, but detail what is wrong. Include a list of what was expected. Remember report one problem at a time, don't combine bugs in one report.

Supporting documentation:

If available, supply documentation. If the process is a report, include a copy of the report with the problem areas highlighted. Include what you expected. If you have a report to compare against, include it and its source information (if it's a printout from a previous version, include the version number and the dataset used)

This information should be stored in a centralized location so that Developers and Testers have access to the information. The developers need it to reproduce the bug, identify it and fix it. Testers will need this information for later regression testing and verification.

Organization

Organization is one of the most important tools available. If your reporting process is organized and standardized it will serve you well. Take the time to develop a standardized method of reporting and train Testers, QA and Beta-testers in its use.

If at all possible, use a tracking system for your defect/development tracking and make sure that everyone using it understands the fields and their importance.

Document your data samples to match up with the bugs/defects reported. These will be useful both to development when fixing the bug and to Testing/QA when it comes time for regression testing.

Summary

A bug report is a case against a product. In order to work it must supply all necessary information to not only identify the problem but what is needed to fix it as well.

It is not enough to say that something is wrong. The report must also say what the system should be doing.

The report should be written in clear concise steps, so that someone who has never seen the system can follow the steps and reproduce the problem. It should include information about the product, including the version number, what data was used.

The more organized information provided the better the report will be.