Department of Electrical and Computer Engineering and Computer Science
Milwaukee School of Engineering
Winter Quarter, 2013-2014
SE-4831: Software Quality Assurance

SE-4831: Software Quality Assurance
Lab 6: Analyzing Eclipse, Apache and Linux Kernel Bug Reports

1. Key Lab Activities

- Historically analyze a set of bug reports to detect bug discovery trends
- Analyze the distribution of bugs within a large data set

2. Introduction

In class, we have talked extensively about different techniques to analyze bugs. In specific, we have looked at trends which can be used to establish that software is ready for release, such as the zero bug bounce and S curves. Our goal for this lab is to analyze a set of software bugs to try and figure out what information we can learn.

The bug data available on the website is collected from three different major projects. For the Apache and Eclipse projects, the bug data that is available represents all bugs which were filed between 2008 (for Apache) and 2011 (for Eclipse) and the present. The Linux bugs represent the complete set of bugs tracked using the Bugzilla tracking systems from project inception. These bugs have been directly exported from the Bugzilla tool and only been modified to generate the XML file and the Access database that is on the course website.

Bugs in the BugZilla system have the lifespan shown in Figure 1. In essence, any bug report which is filed starts in the unconfirmed state. As a bug is confirmed to be a problem, it is placed into the new state. From the new state, it is either assigned or marked resolved. When resolved, it is categorized either as Fixed, duplicate, won't fix, Works for me (essentially not reproducible), or invalid. From a bug analysis standpoint, we are only interested in the bugs which are resolved (either RESOLVED, VERIFIED, or CLOSED) and are Fixed. Fixed indicates that a developer has made a fix to the program. Other states, such as duplicate, merely indicate multiple people found the bug.

In this lab, your goal is to use the tools you have been provided to try and analyzing one set of bug reports. The analysis should include information about the bug rates, the bug discovery curves, etc. You'll also want to try and discover what types of bugs are prevalent and what components tend to be the buggiest pieces. You'll also want to look at how long bugs are open. Is there a relationship between severity and length of time that bugs are open? Are certain components buggier, or are there more problems with certain operating systems than other operating systems? What percentages of bug reports are erroneous? What percent of bug reports are duplicates of other bugs in the system?

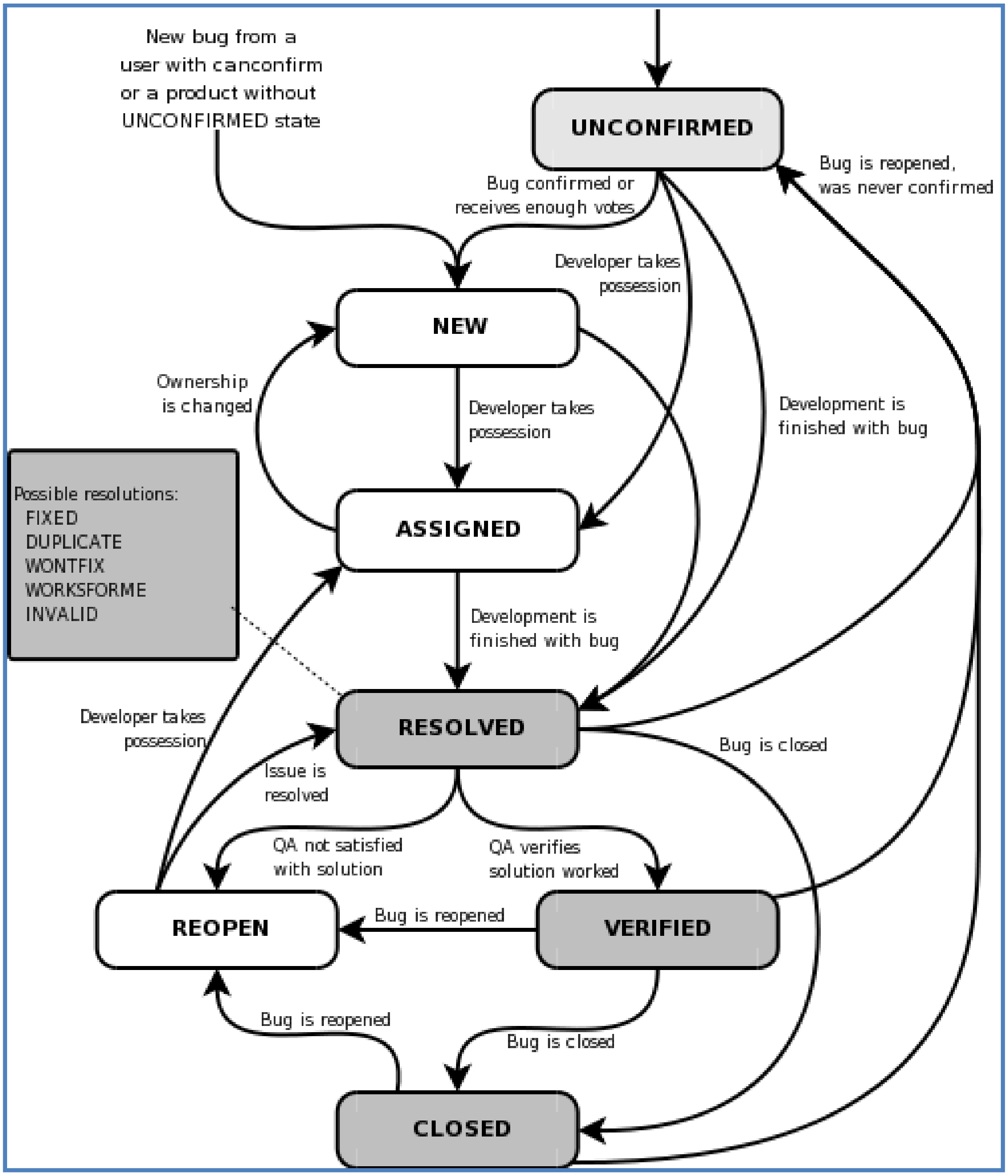
New bug from a user with canconfirm or a product without UNCONFIRMED state
UNCONFIRMED
Bug is reopened, was never confirmed
Bug confirmed or receives enough votes
Developer takes possession
NEW
Ownership is changed
Developer takes possession
Development is finished with bug
ASSIGNED
Possible resolutions:
FIXED
DUPLICATE
WONTFIX
WORKSFORME
INVALID
Development is finished with bug
Developer takes possession
RESOLVED
Bug is closed
Issue is resolved
QA not satisfied with solution
QA verifies solution worked
REOPEN
Bug is reopened
VERIFIED
Bug is reopened
Bug is closed
CLOSED

Figure 1 BugZilla State model

.

3. Deliverables

As a team, you are to deliver a report which provides an analysis of what you have uncovered regarding the bugs. Ideally this should include some form of net bug plots, open bug plots, etc. It may also be useful to look into how long different levels of bugs are active versus the priority, component, etc. How you go about doing this is completely up to you. The data is being distributed as an Access database as well as a RAW XML format, but you are free to pull it into any database or other tool that you may be familiar with.

3.1. Report (One per Team)

Each team should submit a report with the following information

1. Introduction -> What are you trying to accomplish with this lab? What are its goals and objectives?
2. What are your findings -> Include the graphs that you generated or any other supporting analysis. Make certain that you write up an appropriate explanation for your method of analyzing the data as well as findings.
3. Data discussion -> Based on your findings, how consistent is the data that is entered into the bug databases? Do the bug reports convey meaningful information?
4. Things gone right / Things gone wrong -> This section shall discuss the things which went correctly with this experiment as well as the things which posed problems during this lab.
5. Conclusions -> This section shall discuss what has been learned from this laboratory experience. Also indicate those things which you may do differently based on this lab experience.

Reports should be submitted electronically through the web script by 23:59 on January 28, 2014.