How Long Did It Take To Fix Bugs?

Sunghun Kim, E. James Whitehead, Jr.
University of California,
Santa Cruz, CA, USA
{hunkim, ejw}@cs.ucsc.edu

ABSTRACT
The number of bugs (or fixes) is a common factor used to measure the quality of software and assist bug related analysis. For example, if software files have many bugs, they may be unstable. In comparison, the bug-fix time—the time to fix a bug after the bug was introduced—is neglected. We believe that the bug-fix time is an important factor for bug related analysis, such as measuring software quality. For example, if bugs in a file take a relatively long time to be fixed, the file may have some structural problems that make it difficult to make changes. In this report, we compute the bug-fix time of files in ArgoUML and PostgreSQL by identifying when bugs are introduced and when the bugs are fixed. This report includes bug-fix time statistics such as average bug-fix time, and distributions of bug-fix time. We also list the top 20 bug-fix time files of two projects.

Categories and Subject Descriptors
D.2.7 [**Software Engineering**]: Distribution, Maintenance, and Enhancement – *Restructuring, reverse engineering, and reengineering,* D.2.8 [**Software Engineering**]: Metrics – *Product metrics,* K.6.3 [**Management of Computing and Information Systems**]: Software Management – *Software maintenance.*

General Terms
Management, Measurement

1. INTRODUCTION
The number of bugs is commonly used to measure software quality. For example, if a file has 100 cumulative bugs over its development history, we may assume the file is more instable than one that had no bugs in its history. We believe that both bug counts and bug-fix times are important factors for bug related analysis. We can determine the bug-fix time by identifying bug-introducing changes (fix-inducing changes [5]) and corresponding bug fixes. The bug-fix time can be used to measure software quality. For example, if bugs in a software file take a long time to be fixed, it may indicate the file is instable or we need to pay more attention to the file.

We compute the bug-fix time of two open source projects, ArgoUML (period 1/2002 - 3/2003) and PostgreSQL (period 07/1996-11/2000), and report bug-fix time statistics. Our goal is to demonstrate how bug-fix time can be used as a factor for bug related analysis.

MSR '06, May 22-23, 2006, Shanghai, China.

2. EXPERIMENT SETUP
To compute bug-fix time, we need to identify bug-introducing changes and their corresponding fixes, and then measure the time between them. For example, suppose a bug was introduced (in file '*foo*') at revision 3 and it was fixed at revision 9 as shown in Figure 1. We compute the bug-fix time by subtracting the commit time of revision 3 from that of revision 9.

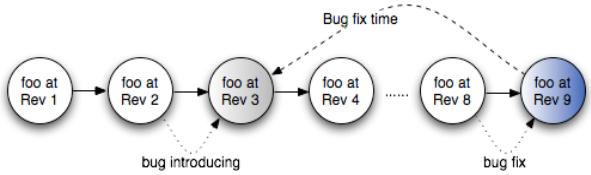

Figure 1. Bug-fix time example.

We first extract change histories of the two projects using the Kenyon infrastructure [1]. We next identify bug fixes by mining change logs. There are two ways to identify a bug-fix: searching for keywords such as "Fixed" or "Bug" [4] and searching for references to bug reports like "#42233" [2, 3, 5]. We use the keyword-based change log search to identify bug fixes. We identify bug-introducing changes by applying the fix-inducing change identification algorithms described in [5]. We then obtain the commit time of the identified bug-introducing changes and their corresponding bug fixes from project histories. From the commit times, we compute each bug-fix time and the average bug-fix time of each file.

3. BUG-FIX TIME
In this section we report bug-fix time statistics of two projects.

3.1 Bug Numbers and Fix Time
We show the distribution of bug counts for each bug-fix time in Figure 2 and Figure 3. Bug fixes times in buggy files range from 100-200 days (the spikes in Figure 2 and Figure 3).

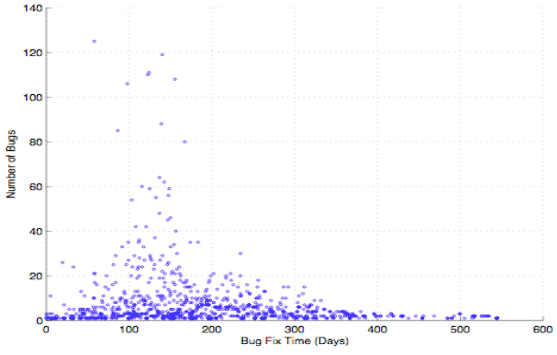

Figure 2. Distributions of bug counts by bug-fix time of ArgoUML.

Table 1. Top 20 files with greatest bug-fix times

Rank	ArgoUML Files	Bug fix time (days)	Bug count	PostgreSQL Files	Bu fix time (days)	Bug count
1	argouml/src_new/org/argouml/uml/ui/UMLInitialValueComboBox.java	332	9	pgsql/src/backend/commands/define.c	504	19
2	argouml/src_new/org/argouml/uml/ui/UMLAttributesListModel.java	328	6	pgsql/src/backend/access/rtree/rtree.c	482	14
3	argouml/src_new/org/argouml/ui/NavigatorConfigDialog.java	324	9	pgsql/src/backend/utils/hash/dynahash.c	474	17
4	argouml/src_new/org/argouml/kernel/ProjectMember.java	320	7	pgsql/src/backend/utils/cache/inval.c	472	16
5	argouml/src_new/org/argouml/uml/ui/UMLTaggedBooleanProperty.java	318	7	pgsql/src/include/storage/bufpage.h	450	14
6	argouml/src_new/org/argouml/uml/ui/ActionSaveGraphics.java	317	6	pgsql/src/backend/utils/cache/relcache.c	444	84
7	argouml/src_new/org/argouml/uml/ui/UMLMultiplicityComboBox.java	317	6	pgsql/src/backend/catalog/pg_proc.c	425	18
8	argouml/src_new/org/argouml/uml/cognitive/critics/WizAssocComposite.java	315	6	pgsql/src/backend/optimizer/path/allpaths.c	422	37
9	argouml/src_new/org/argouml/ui/FindDialog.java	312	7	pgsql/src/backend/executor/nodeMergejoin.c	419	17
10	argouml/src_new/org/argouml/uml/DocumentationManager.java	312	15	pgsql/src/backend/utils/fmgr/dfmgr.c	408	17
11	argouml/src_new/org/argouml/uml/ui/ActionNew.java	310	12	pgsql/src/backend/commands/trigger.c	408	25
12	argouml/src_new/org/argouml/cognitive/ui/ToDoPerspective.java	306	6	pgsql/src/backend/utils/cache/catcache.c	407	32
13	argouml/modules/php/src/org/argouml/language/php/generator/GeneratorPHP.java	305	11	pgsql/src/backend/utils/init/postinit.c	399	46
14	argouml/src_new/org/argouml/uml/cognitive/critics/CrNameConflict.java	305	6	pgsql/src/backend/executor/nodeHash.c	393	19
15	argouml/src_new/org/argouml/uml/ui/UMLComboBoxEntry.java	304	6	pgsql/src/backend/executor/nodeAgg.c	391	53
16	argouml/src_new/org/argouml/cognitive/critics/ui/CriticBrowserDialog.java	304	8	pgsql/src/backend/rewrite/rewriteDefine.c	385	29
17	argouml/src_new/org/argouml/uml/ui/ActionAddOperation.java	292	15	pgsql/src/backend/access/gist/gist.c	384	19
18	argouml/src_new/org/argouml/uml/ui/ActionDeleteFromDiagram.java	289	10	pgsql/src/backend/nodes/readfuncs.c	382	60
19	argouml/src_new/org/argouml/uml/ui/ActionAddTopLevelPackage.java	289	6	pgsql/src/backend/catalog/pg_type.c	376	22
20	argouml/src_new/org/argouml/language/ui/SettingsTabNotation.java	287	15	pgsql/src/backend/commands/rename.c	376	24

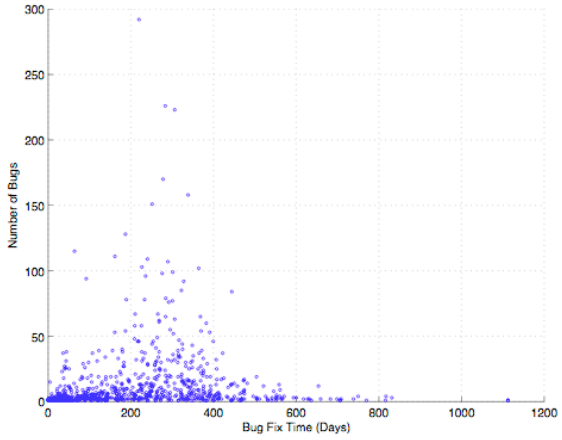

Figure 3. Distributions of bug counts per bug-fix time of PostgreSQL.

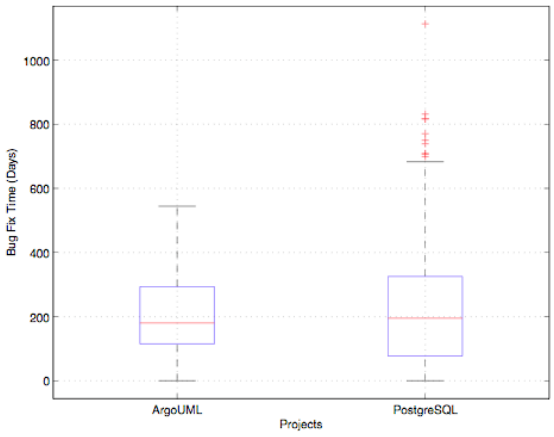

Figure 4. Bug-fix time (days) of the two projects. Two boxes indicate 50% of bug-fix time (25% to 75% quartile). The middle line in boxes indicates the median value of bug-fix time.

Figure 4 shows the bug-fix time of the two projects using box plots. They show that fixing 50% of the bugs requires appx. 100 to 300 days (the two boxes in Figure 4). The median bug-fix time is about 200 days.

3.2 Number and Bug-fix Time

Table 1 lists the top 20 files with greatest bug-fix times, whose bug counts are greater than average. The listed files may need attention to determine why bug fixes take such a long time and may need to be refactored to permit faster bug fixes in the future .

4. CONCLUSION

By mining software histories of two projects, ArgoUML and PostgreSQL, we computed and analyzed the bug-fix time of each file. We believe that bug-fix time is useful, and should be widely used for bug related analysis.

5. REFERENCES

[1] J. Bevan, E. J. Whitehead, Jr., S. Kim, and M. Godfrey, "Facilitating Software Evolution with Kenyon," Proc. of the 2005 European Software Engineering Conference and 2005 Foundations of Software Engineering (ESEC/FSE 2005), Lisbon, Portugal, pp. 177-186, 2005.

[2] D. Cubranic and G. C. Murphy, "Hipikat: Recommending pertinent software development artifacts," Proc. of 25th International Conference on Software Engineering (ICSE), Portland, Oregon, pp. 408-418, 2003.

[3] M. Fischer, M. Pinzger, and H. Gall, "Populating a Release History Database from Version Control and Bug Tracking Systems," Proc. of 2003 Int'l Conference on Software Maintenance (ICSM'03), pp. 23-32, 2003.

[4] A. Mockus and L. G. Votta, "Identifying Reasons for Software Changes Using Historic Databases," Proc. of International Conference on Software Maintenance (ICSM 2000), San Jose, California, USA, pp. 120-130, 2000.

[5] J. Sliwerski, T. Zimmermann, and A. Zeller, "When Do Changes Induce Fixes?" Proc. of Int'l Workshop on Mining Software Repositories (MSR 2005), Saint Louis, Missouri, USA, pp. 24-28, 2005.