When Do Software Issues and Bugs get Reported in Large Open Source Software Project?

Rakesh Rana and Miroslaw Staron

Department of Computer Science & Engineering
Chalmers/ University of Gothenburg
Göteborg, Sweden
rakesh.rana@gu.se

Abstract. In this paper we examine the reporting pattern of more than 7000 issue reports from five large open source software projects to evaluate two main characteristics: (1) when do defects get reported - does there exist any distinct patterns, and (2) is there any difference between reported defect inflow and actual defect inflow for these projects. The two questions evaluated here are important from practical standpoint. Detailed knowledge of specific patterns in defect reporting can be useful for planning purposes, while differences between reported and actual defect inflow can have implications on accuracy of dynamic software reliability growth models used for defect prediction/reliability assessment. Our results suggest that while there exist distinct variation over when defects are reported, the ratio of reported to actual defects remains fairly stable over period of time. These results provide more insights into possible group of people who contribute to OSS projects and also enhance our confidence in applying SRGMs using reported defect inflow. Our test with logistic growth model show that using reported bugs provided asymptote (total expected bugs) predictions that were on average only 4.8% different than using actual bugs for making such predictions.
Keywords Software defects, Issue reporting, Mining software repositories, Bug reports, Data quality, Open source software, SRGMs.

1 Introduction

An important source of information in proprietary and open source software projects is the issue tracking systems. A large part of development and maintenance activities can be tracked and traced to reports from issue tracking systems. While there has been much effort devoted to find out who actually contributes to OSS system development and maintenance [1] [2], little is known about when such contributions are made. Understanding when issues are reported can provide useful insights into the characteristics of OSS contributors. Śliwerski et al. [3] analysed bug database of MOZILLA and ECLIPSE to investigate fix induced changes, one of their interesting observation was that the likelihood that a change will induce a fix was highest on Friday. We use the time stamp of all issues to analyse when issues and actual bugs are reported for five studied OSS projects.

Another important aspect related to using information from OSS issue tracking systems is the quality of data, which has been a topic of much research [4] [5] [6]. While in closed-source projects issue tracking may be strictly used only for recording and managing bugs, in open source projects issue tracking systems usually tend to serve as interface for discussion between developers and users. Thus often apart from listing bugs, issue tracker of OSS projects contain entries requesting new features, perfective or preventive maintenance, architectural discussions, need for re-factoring, etc. [7]. Also usually different kinds of issues submitted to OSS issue tracking systems are simply labelled "Bug" which can introduce potential bias to defect prediction models as highlighted in study by Herzig et al. [8].

Software Reliability Models (SRM) are defined as *"mathematical expression that specifies the general form of the software failure process as a function of factors such as fault introduction, fault removal, and the operational environment."* [9]. SRGMs can be classified as white box models referring to models that use source code metrics for making assessment/prediction of defect proneness of given software artefact - these have been shown to be affected due to misclassification of non-bugs as bugs [8]. The other class of SRM are black box models usually refereed to as Software Reliability Growth Models (SRGMs) that only uses bug/defect inflow data to make prediction of future defects/reliability growth of given software system [10] [11] [12]. While the effect of misclassification on white box models has been studied, it is not clear if such misclassification can also affect the use of dynamic models (SRGMs). Since black box models only use the trend of number of bugs reported for modelling reliability growth they can provide satisfactory results as long as the misclassification ratio is stable i.e. percentage of issues misclassified as bugs remains fairly stable. We analyse the two ratios for investigating this - ratio of actual bugs to total issues, and ratio of actual bugs to reported bugs. Our results suggest that while there exist large variation in total issues and bugs being reported over day, week and months - the ratio of actual bugs to total issues and reported bugs remains much more stable.

The rest of paper is organised as follows, the next Section (2) describes the data used in this study. Section (3) provides an overview of background and related work, followed by section 4 outlining the results from the study. In section 5 we test the effect of misclassification of issues on application of SRGMs which is followed by conclusion of paper and future work in the section 6.

2 The Data

We studied five open source Java projects with active development and since these projects are developed and maintained by APACHE and MOZILLA community they also tend to follow a strict bug reporting and fixing process. The main reason of choosing these projects was also the availability of manual classification of issues for the studied period, Herzig et al. [8] examined and manually

classified the issue reports from these projects to study the rate of misclassification and resulting bias on bug prediction models. In this study we examined all issue reports in the same period for analysing when do issues and bugs actually get reported and if misclassification can also introduce bias in SRGM models. The details of project and study period are summarised in table 1.

Table 1. Table with OSS project details and studied time period

OSS Project	Time Period	Maintainer	No of Issues	Ref.
HttpClient	11/2001-04/2012	APACHE	746	Website[1]
Jackrabbit	09/2004-04/2012	APACHE	2402	Website[2]
Lucene-Java	03/2004-03/2012	APACHE	2443	Website[3]
Rhino	09/1999-02/2012	MOZILLA	584	Website[4]
Tomcat5	05/2002-12/2011	APACHE	1226	Website[5]

To collect the data, we first mined all issue reports from the five OSS projects including all details (issue id, time stamp, original classification, description etc.), we then matched each issue id to data on manual classification of Herzig et al. [8] made publicly[6] available by the authors. We then used the timestamp information of all issues to analyse when issues were reported by the individual contributors and utilized original and manual classification to check the trend of actual bugs to reported bugs.

3 Background and Related Work

Software issues, defects and bugs are frequently used in software literature and this paper, we distinguish between them as follows:
Term software issue is used to refer to a report filed by users or developers into the given OSS projects' issue database. These issues can be further classified as Defects/Bugs, request for enhancements, improvement requests, documentation, refactoring requests, etc. [8]. Defect and Bug are used interchangeably in this paper referring to issues that require a corrective maintenance tasks usually achieved by making semantic changes to the source code.

In the OSS development, a large percent of issue reports generally gets reported by the users. The core member of the OSS team usually review these issue reports and identify which of these report as valid, invalid, require corrective maintenance (defect/bug) and so on. If the described defects are valid, the

[1] https://hc.apache.org/httpcomponents-client-ga/
[2] https://jackrabbit.apache.org/jcr/index.html
[3] https://lucene.apache.org/
[4] https://developer.mozilla.org/en-US/docs/Mozilla/Projects/Rhino
[5] http://tomcat.apache.org/
[6] http://www.softevo.org/bugclassify

core team will start to repair it or adopt the patches provided by the users. The terms used frequently in this context here are:

- **Total issues:** The total number of issues reported by all users and contributors of a given OSS project.
- **Reported bugs:** Number of issues that are indicated (flagged/classified) as a Bug by the individual filing/reporting the given issue.
- **Actual bugs:** refer to number of issues that are identified as actually bugs using the manual classification of all issues as reported by Herzig et al. [8].

Kim et al. [13] highlighted the impact of noise in defect data on defect prediction models. The authors list guidelines for acceptable noise level and also propose algorithm for noise detection and elimination to address the problem. Ramler and Himmelbauer [14] also studied the effect of noise (misclassification) in bug reports data on the predictive accuracy of defect classification models for an industrial software system. They investigated the causes of noise in bug report data and experimented the effect of noise level on the performance of defect classification models. In this paper we investigate if noise can also effect application of software reliability growth models which are black-box models used for predicting the shape of defect inflow and total expected defect in the context of software defect predictions. Misclassification of failures can also lead to implications to effectiveness of verification and validation (V&V) as highlighted in study by Falessi et al. [15]. Knowing in which periods a project is likely to have large inflow of issues/defects can help OSS core team to plan and allocate their resources effectively helping them to minimize classification errors leading to enhanced V&V effectiveness.

Earlier studies by Staron and Meding [16][17] emphasized the importance of predicting short-term defect inflow for project planning and monitoring purposes in large industrial software projects. The short term and weekly defect inflow prediction models proposed in their work are intended to support project mangers towards helping them make more accurate resource allocation decisions. While the earlier work has been evaluated in the context of large industrial software projects, in the paper we look closely into the OSS projects. By understanding when most issues (and actual bugs) are reported over time and effect of noise in defect data on total defect count predictions, this research is aimed at helping OSS core members to manage their project more effectively. Close examination of patterns of software issues inflow over time also leads us towards better understanding of OSS project contributors.

4 Results

4.1 Issues inflow profile

First we examined the inflow frequency of reported issues for the five open source projects. Figure 1 provide the number of issues reported weekly and also the cumulative issues inflow profile over the studied period. A detailed inspection of issues inflow can provide insights into the software development/maintenance

process, help with planning resource/work delegation. Further issues/defect inflow profile can also be used to model and thus predict the expected number of issues inflow in future or model the reliability characteristics of software product using modelling techniques such as software reliability growth models (SRGMs) [10].

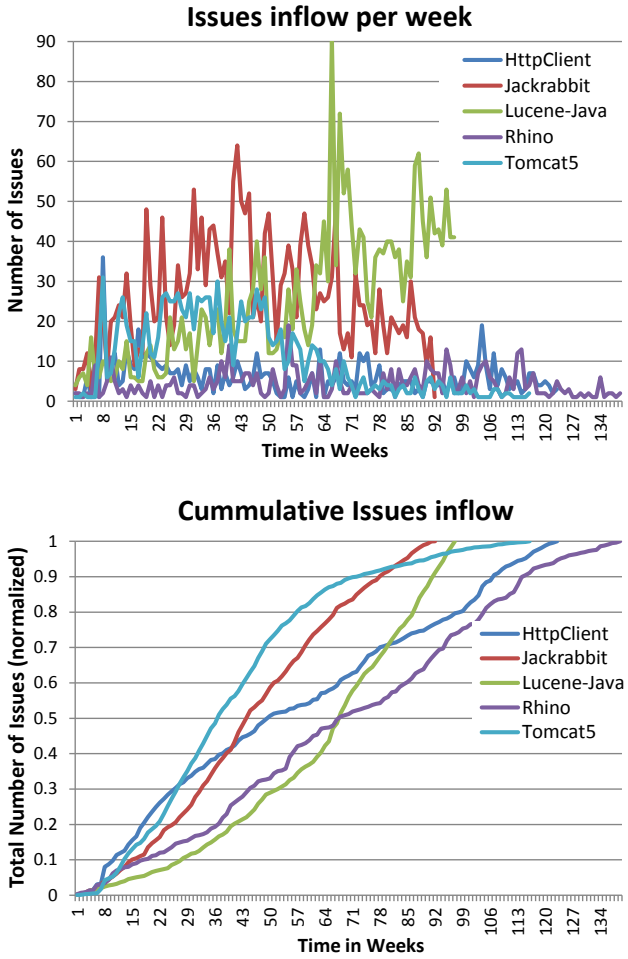

Fig. 1. Total issues inflow per week and cumulative inflow profile

From figure 1, we note that issues inflow varies widely (not surprising) for different OSS projects. What is interesting here are the differences in the pattern of weekly defect inflow which are also evident from the cumulative defect inflow, we observe that.

1. The issue intensity (number of reported issues per time unit) for OSS project HttpClient and Rhino remained pretty low and stable for the studied period. The cumulative issues profile for such projects is more or less a linear in nature which has been associated with incremental development and maintenance activities.
2. The issues inflow of Jackrabbit project show a start with low issues intensity, which maximizes in the middle followed by the fall in issues intensity. The issues inflow pattern observed for this project is associated with S-shaped issues inflow (see cumulative issues profile) and indicates lower issues inflow as the associated product matures.
3. In case of Lucene-Java project, the issues intensity has an increasing trend that translates into an convex shaped issues inflow profile. Such issues inflow can either result from rapid development and deployment/adoption and needs more resources for managing the issues inflow increase with time.
4. The OSS project Tomcat5 issues inflow starts with high issues intensity in the early stages of studied period and which falls substantially as the time progress. Such pattern produces a concave issues profile where majority of issues are reported early in the project and as these issues are resolved and product matures the reported issues intensity decreases. The small issues intensity at late phases can also be a result of new release of given product.

Thus examination of issues inflow of different OSS projects revel that their reported issues inflow can be quite different from one another given number of factors that affect the project such as pace of development, active community, adoption, maturity, and release management. Nonetheless understanding of issues inflow of given OSS project release can not only enhance our knowledge of these factors for given OSS project, but can also help OSS communities to plan for their following releases.

There have been reports that had questioned the quality of defect/bug reports from OSS projects [8] [4]. These studies suggest that about 40% of files marked as defective in reality does not have any associated bugs [8]. While misclassification of defects can pose a serious threat to defect prediction/classification models trying to predict which files/modules are defect prone, such misclassification may not pose a serious risk to application of SRGMs which model reliability on the trend of reported defects as long as the ratio of reported defects to actual defects is stable over time.

4.2 Issues reported on monthly basis

From figure 2, we observe that total issues reported, issues reported that were classified as bugs (reported bugs) and issues that were actually bugs (as per manual classification from [8] study - actual bugs) is fairly stable for each month of the year. Total issues and reported bugs were lower than their mean value for month of December which is understandable due to public holidays, but surprisingly the actual reported bugs were close to their mean value. One reason for this observation could be due to the result of larger use of particular OSS products leading to discovery of real bugs that counter the expected depression of

actual bug reports from the holiday season in the given month. It is also observed that total reported issues, reported and actual bugs register growth for the first three months of the year and lower than mean values for the summer months (April to July). The depression in the summer months period be partly explained by onset of busy (exam) periods in Universities, end of financial year/quarter over many countries that could put higher pressure on OSS project contributors that work full-time for private companies, and the associated vacation period.

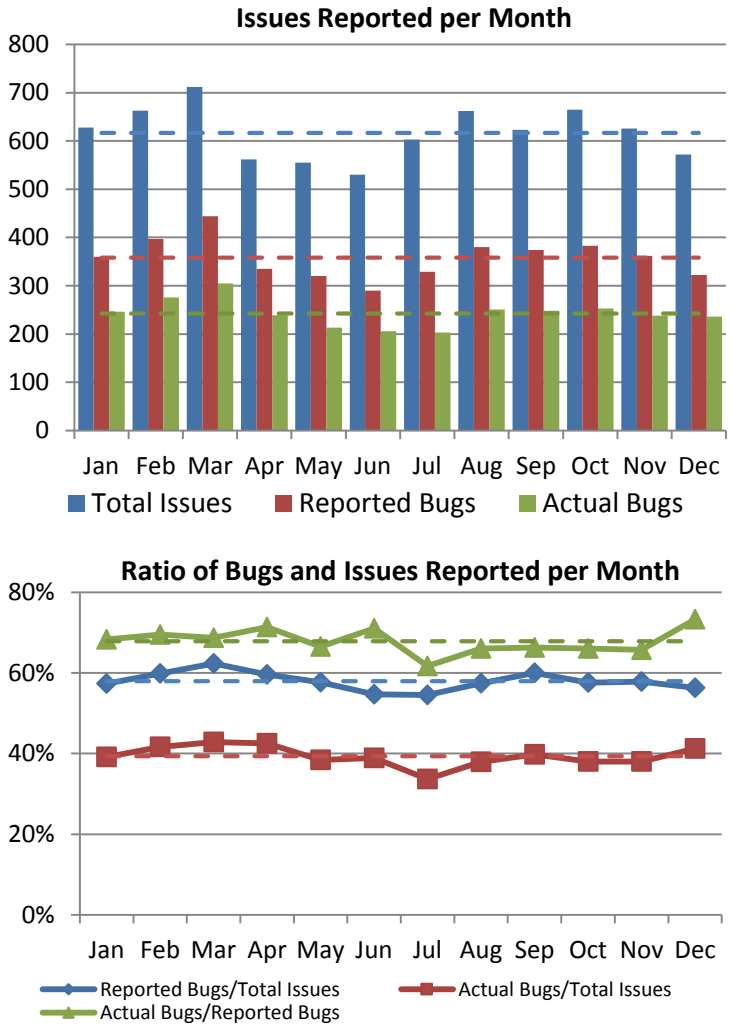

Fig. 2. Monthly distribution of total issues, reported bugs and actual bugs over the five OSS projects

Taking a closer look at what proportion of total reported issues are marked as Bugs (by the issue reporter) and proportion of actual bugs to the total and reported bugs provide a fairly stable trend. About 58% of all reported issues are classified as bugs by the original reporter, this ratio does not vary much although a small trend that follows the total issues reported is observed where slightly higher percentage of all issued reported in early months of year are reported bugs, while the ratio is lower than its mean value for the summer period. With respect to actual bugs to reported bugs, which is an important ratio to know if application of SRGMs using reported bugs could produce misleading results - the trend is quite linear with about 68% of reported bugs were actual bugs over the period of time. The two months that had different values of this ratio were July and December when the ratio of actual to reported bugs were approx. 62% and 73% respectively. Except for these two months the actual to reported bugs ratio stays quite close to its mean value and thus the probability of large impact on the SRGMs models that use reported bugs is likely to be low.

4.3 Issues reported on weekly basis

The trends of issues and bugs reporting we saw on monthly basis (figure 3) can be examined in more details by breaking them down on weekly basis. From figure **??**e observe that the total issues, reported and actual bugs start to fall rapidly in last three weeks of year and recovers close to its mean value just in the second week of new year. Week 2-8 usually register close to mean value of issues and bug reports while it stays much higher than mean for the next 5 weeks. Over weeks 14-32, total issues, reported and actual bugs were almost consistently lower than their respective mean values.

With regard to the ratio of actual bugs to total issues and reported bugs, the trend stayed close to its mean value with some fluctuations over weeks. But given the variation of these ratio is not too high, it is safe to interpret that using total reported issues or reported bugs as a proxy for actual bugs is likely to work for most practical purposes such as projection of how many actual bugs may be needed to handle next month etc.

4.4 Issues reported by week day

There have been many studies trying to figure out who really contributes to the development of OSS projects, but little is known about when do these contributions occur. We examined about 7000 issues from the five OSS projects to see when do they get reported.

While it is not too radical to expect that most OSS issue reports may be reported over the weekends when contributors who have full time education/jobs have spare time to work on their hobby projects - the empirical data suggest otherwise. On average majority of issues, reported and actual bugs are accounted during the work week. The pattern seen here (greater than mean reports originating on Tue-Sat, while lowest number of issues and bugs being reported on

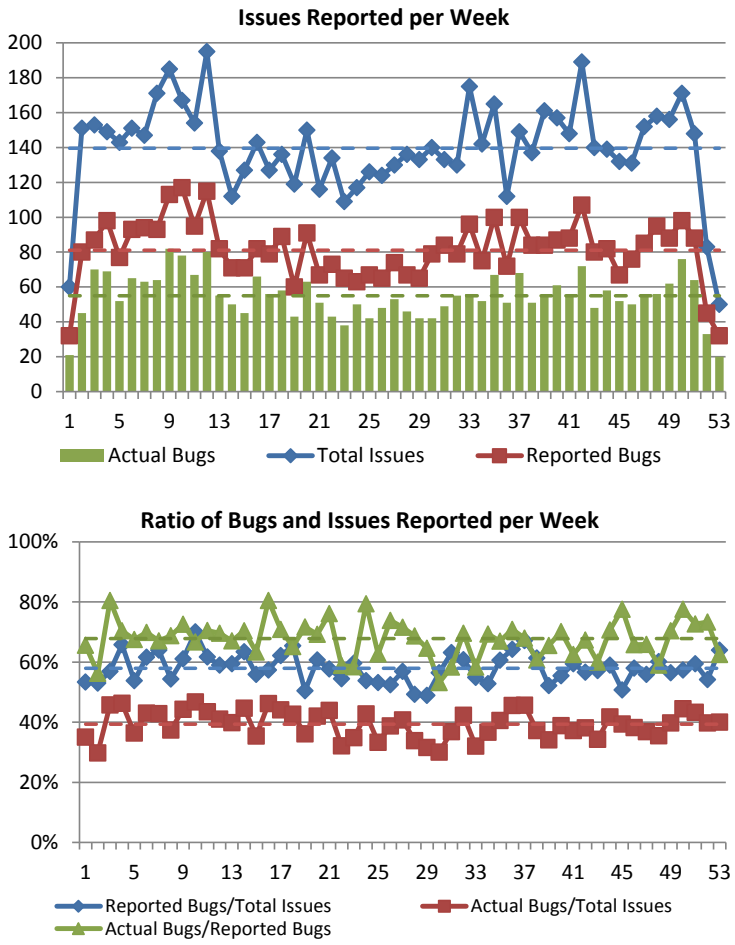

Fig. 3. Weekly distribution of total issues, reported bugs and actual bugs over the five OSS projects

Sunday and Mondays) indicate towards majority of OSS contributors coming from individuals with full time studies/jobs.

The following characteristics could have contributions to the observed distribution of issues reporting for OSS projects:

1. The first day of week (Mon), most contributor are busy at their primary task (education/job) and do not find enough time to contribute to the OSS project.
2. As the week progress (Tue-Fri), the number of total issues and bugs reported increases as a large proportion of contributors probably works alongside their primary duties of education/jobs.

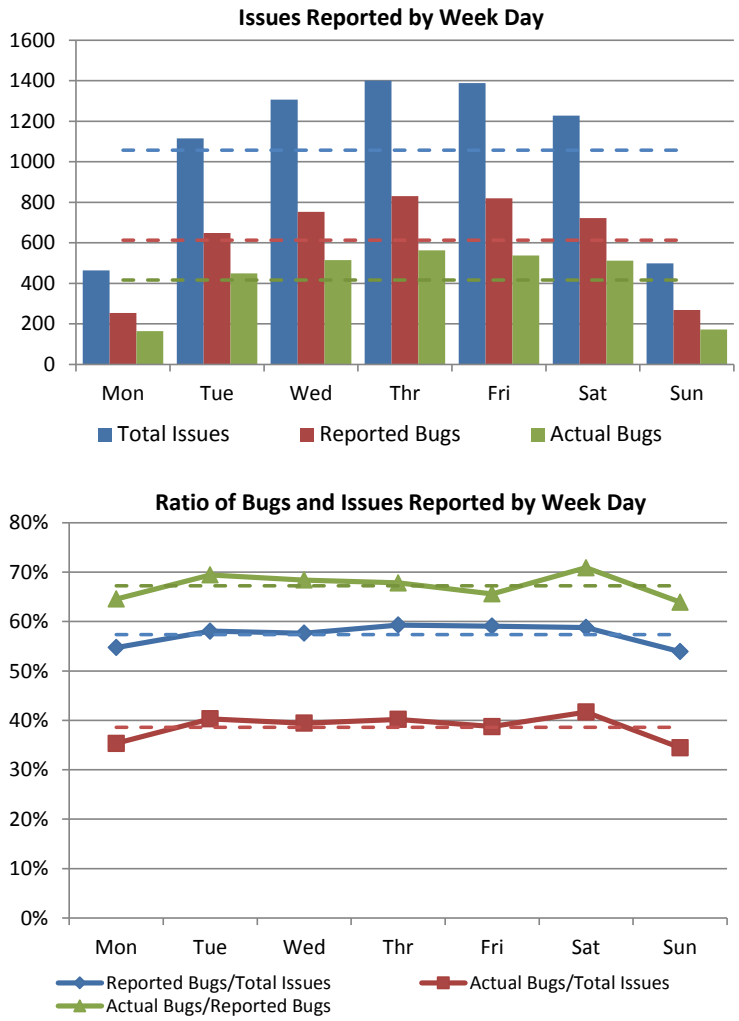

Fig. 4. Distribution of total issues, reported bugs and actual bugs over the five OSS projects by week day

3. On Saturday which is the first week day off in most western countries following 5-day work week, while there is drop in absolute number of issues compared to peak of Thr/Friday, but still a large number of issues and bugs are registered (over how many are reported on Tuesday). This suggest that OSS contributors may be using the momentum of work week and utilizing the free time available on the day off to contribute to their choice of OSS project.
4. Finally on Sunday, it seems most contributors take time off before the next week start to refresh themselves before staring a new work week.

And while the ratio of actual bugs over total issue and reported bugs is quite stable over week days, interestingly on Sunday and Mondays as there are lower number of issues and bugs reported they also tend to be slightly less inclined to be referring to actual bug into the system than on other days of week.

5 Does misclassification of issues lead to bias in SRGM predictions?

Next to empirically evaluate if using reported bugs as proxy for actual bugs can lead to bias results or not when using dynamic/black box software reliability models. We applied Logistic model to data on reported and actual bugs from the five projects, the Logistic model was chosen as it has been shown to provide high predictive accuracy in our earlier studies [10]. Logistic model is an S-shaped model [18] with mean value function given by equation 1, where a represents the asymptote, b, the growth rate of curve, and c is the constant.

$$m(t) = \frac{a}{(1 + e^{-b(t-c)})} \tag{1}$$

For the experiment following steps were performed:

1. We first took the actual bug inflow data (manually classified bugs), logistic model was applied on the 90% of data points and model parameters were noted.
2. Then we used reported bug inflow data (issues classified as bug by original contributor, i.e. data with misclassification noise) and used same model again on 90% of data points and model parameters were recorded.
3. The asymptote obtained from using reported bugs was then adjusted to reflect the proportion of actual bugs out of reported bugs for the given project. This number can be easily obtained for an OSS project by manually classifying a sample of user classified bug reports as follows. Assuming we manually analysed a sample s of issues reported (flagged) as bugs by the issue reporter, but only $m, (m \leq s)$ were found to be actual bugs on manual examination. Then the asymptote obtained from using reported bugs can be adjusted for actual bugs as:
   Adj Predicted Asymptote $= \frac{s}{m} *$(Predicted Asymptote using reported bugs).
4. Finally we compare the parameters (Asymptote, growth rate) obtained from using reported bugs to model parameters obtained from using actual bug inflow data. The results are summarized in table 2.

As it can be noted from the table 2, the SRGM (in this case logistic model) parameters obtained using reported bug inflow are close to parameters using actual bug inflow data. While these exist some variation across the projects, in general we observe that using reported bug inflow can provide actual bug inflow profile with good accuracy without actually need for manual classification of all issues. The relative error between asymptote prediction obtained using reported bugs is close to asymptote prediction obtained using actual bugs which require

Table 2. Table of parameters obtained by applying logistic SRGM on reported bugs (proxy) and actual bugs

OSS Project	Asymptote (a)			Growth rate (b)		Constant term (c)	
	Reported bugs (Adj)	Actual bugs	Relative Error	Reported bugs	Actual bugs	Reported bugs	Actual bugs
HttpClient	261	260	0.4%	0.051	0.050	39.7	41.3
Jackrabbit	901	923	-2.4%	0.069	0.065	42.3	44.6
Lucene-Java	741	685	8.2%	0.049	0.054	67.6	65.7
Rhino	338	301	12.3%	0.035	0.040	76.4	64.5
Tomcat5	644	640	0.6%	0.082	0.084	36.7	33.3

manual classification of each reported issue. The average relative error (magnitude) over five projects was 4.8% and thus provided good proxy for predicting expected number of total defects using the readily available data on reported issues and bugs.

6 Conclusions and Next Steps

In this paper we examined two important aspects of open source software development. Using more than 7000 issue reports from 5 large OSS projects. We also closely inspected when the issues and actual bugs are reported for the OSS projects. The most productive periods for issue and bug reporting was identified as periods between Jan-Mar and Aug-Nov, while the activity over December and summer months being lower than average. Another interesting observation made was that activity with respect to issues and bug reporting was at its minimal on first (Monday) and last day (Sunday) of week, while maximum occurred at middle of week on Thursday. A large proportion of issues and bug reports are also filed on Saturday which can be indicative of individuals contribution to OSS projects using their free time.

There have been concerns raised about data quality of issue reports for OSS projects, specifically regarding the misclassification of issues as bugs when they were not bugs in reality. Misclassification of bugs can pose serious threat to defect prediction/classification studies that aim to detect/predict which files/modules are defect prone. We examined if misclassification can affect also mislead the planning or projections using dynamic software reliability growth models that only uses the defect inflow data. Our results suggest that while number of issues and reported bugs vary over time, the ratio of actual bugs to total issues and reported bugs stay pretty stable over time, thus predictions that uses total issues as proxy for actual bugs and SRGMs using total issues/reported bugs are less likely to be impacted by misclassification compared to models predicting the likelihood of a particular file/module being defective.

In this research, the focus has been on reporting of issues and bugs in OSS projects, the future work in this direction can look into more closely the pattern

of commits that can reveal further information on OSS project contributions. Some of the research questions and directions that can be pursued are:

- Analysis of at what local time these commits and issue reports are made can also help us build better profile of who actually contributes to OSS project and when these contribution occur,
- How much can we learn more about the patterns within reported issues and have better understanding of OSS contributors by using defect classification techniques (for e.g. Orthogonal defect classification [19]) to the issues from OSS bug repositories,
- How does patters of issues and bug reporting differ for cases where OSS projects are managed by government or commercial organizations, etc.

Acknowledgment

The work presented here has been funded by Vinnova and Volvo Cars jointly under the FFI programme (VISEE, Project No: DIARIENR: 2011-04438).

References

1. Mockus, A., Fielding, R.T., Herbsleb, J.D.: Two case studies of open source software development: Apache and mozilla. ACM Transactions on Software Engineering and Methodology (TOSEM) **11**(3) (2002) 309–346
2. Lerner, J., Tirole, J.: Some simple economics of open source. The journal of industrial economics **50**(2) (2002) 197–234
3. Śliwerski, J., Zimmermann, T., Zeller, A.: When do changes induce fixes? ACM sigsoft software engineering notes **30**(4) (2005) 1–5
4. Bettenburg, N., Just, S., Schröter, A., Weiß, C., Premraj, R., Zimmermann, T.: Quality of bug reports in eclipse. In: Proceedings of the 2007 OOPSLA workshop on eclipse technology eXchange, ACM (2007) 21–25
5. Runeson, P., Alexandersson, M., Nyholm, O.: Detection of duplicate defect reports using natural language processing. In: Software Engineering, 2007. ICSE 2007. 29th International Conference on, IEEE (2007) 499–510
6. Hooimeijer, P., Weimer, W.: Modeling bug report quality. In: Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering, ACM (2007) 34–43
7. Antoniol, G., Ayari, K., Di Penta, M., Khomh, F., Guéhéneuc, Y.G.: Is it a bug or an enhancement?: a text-based approach to classify change requests. In: Proceedings of the 2008 conference of the center for advanced studies on collaborative research: meeting of minds, ACM (2008) 23
8. Herzig, K., Just, S., Zeller, A.: It's not a bug, it's a feature: how misclassification impacts bug prediction. In: Proceedings of the 2013 International Conference on Software Engineering, IEEE Press (2013) 392–401
9. Standard: Ieee, std 1633-2008, recommended practice on software reliability (2008)
10. Rana, R., Staron, M., Berger, C., Hansson, J., Nilsson, M., Törner, F., Meding, W., Höglund, C.: Selecting software reliability growth models and improving their predictive accuracy using historical projects data. Journal of Systems and Software **98** (2014) 59–78

11. Rana, R., Staron, M., Mellegård, N., Berger, C., Hansson, J., Nilsson, M., Törner, F.: Evaluation of standard reliability growth models in the context of automotive software systems. In: Product-Focused Software Process Improvement. Springer (2013) 324–329
12. Rana, R., Staron, M., Berger, C., Hansson, J., Nilsson, M., Torner, F.: Evaluating long-term predictive power of standard reliability growth models on automotive systems. In: Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on, IEEE (2013) 228–237
13. Kim, S., Zhang, H., Wu, R., Gong, L.: Dealing with noise in defect prediction. In: Software Engineering (ICSE), 2011 33rd International Conference on, IEEE (2011) 481–490
14. Ramler, R., Himmelbauer, J.: Noise in bug report data and the impact on defect prediction results. In: Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2013 Joint Conference of the 23rd International Workshop on, IEEE (2013) 173–180
15. Falessi, D., Kidwell, B., Huffman Hayes, J., Shull, F.: On failure classification: the impact of getting it wrong. In: Companion Proceedings of the 36th International Conference on Software Engineering, ACM (2014) 512–515
16. Staron, M., Meding, W.: Predicting short-term defect inflow in large software projects–an initial evaluation. In: 11th International Conference on Evaluation and Assessment in Software Engineering, EASE. (2007)
17. Staron, M., Meding, W.: Predicting weekly defect inflow in large software projects based on project planning and test status. Information and Software Technology **50**(7) (2008) 782–796
18. Khoshgoftaar, T.M., Allen, E.B.: Logistic regression modeling of software quality. International Journal of Reliability, Quality and Safety Engineering **6**(04) (1999) 303–317
19. Chillarege, R., Bhandari, I.S., Chaar, J.K., Halliday, M.J., Moebus, D.S., Ray, B.K., Wong, M.Y.: Orthogonal defect classification-a concept for in-process measurements. Software Engineering, IEEE Transactions on **18**(11) (1992) 943–956