Open Source Prediction Methods: a systematic literature review

M.M. Mahbubul Syeed, Terhi Kilamo, Imed Hammouda, and Tarja Systa

Tampere University of Technology, Finland
{mm.syeed, terhi.kilamo, imed.hammouda, tarja.systa}@tut.fi

Abstract. For the adoption of Open Source Software (OSS) components, knowledge of the project development and associated risks with their use is needed. That, in turn, calls for reliable prediction models to support preventive maintenance and building quality software. In this paper, we perform a systematic literature review on the state-of-the-art on predicting OSS projects considering both code and community dimension. We also distill future direction for research in this field.

Keywords: Open Source; Systematic Literature Review; Prediction.

1 Introduction

The use of Open Source Software (OSS) is increasingly becoming a part of the development strategy and business portfolio of IT organizations.To adopt an OSS component effectively, an organization needs knowledge of the project development, composition and possible risks associated with its use, due to its unconventional and complex development process and evolution history [1]. This in turn, calls for building reliable prediction models and methods supporting error prediction, measuring maintenance effort and cost of OSS projects.

In this paper, we present a literature Review on prediction studies to analyze OSS projects both from the point of view of the product and the community. The contribution of this work are as follows, (a) a study on the state-of-the-art in OSS prediction methods and approaches; (b) future directions of research work in this field; (c) developed a reusable literature review protocol following the guideline of [2] that can be used as a model for review studies in software engineering. Only key contributions of the work is presented in this paper. Detail discussion on the review process, associated thread to validity and elaborated results with more research questions can be found in *http://literature-review.weebly.com/*.

2 Review methodology

For this review, we developed a review protocol by keeping perfect alignment with the guidelines suggested by Kitchenham [2]. We briefly discussed the review protocol here. A detail discussion on this along with the list of 36 articles reviewed, can be found at *http://literature-review.weebly.com/*.

2.1 Research Questions

For this review, we defined a set of research questions as presented in Table 1.

Table 1. Research Questions

	Research Questions	**Main Motivation**
Q1.	What are the main focuses/purposes of the study?	To identify the focus area of the prediction work (e.g., fault prediction, effort prediction).
Q2.	What are the datasets of OSS projects exploited in prediction?	To identify the data sources of an OSS project that are used for the prediction models.
Q3.	What kinds of methods are used in predicting OSS projects?	Explore the trends and methods used for prediction in the context of OSS.
Q4.	What kinds of metrics are used in predicting OSS projects?	To identify the trend and usage of different types metric suits for prediction in the context of OSS.

2.2 Article Selection

Inclusion criteria. for assessing the suitability of the articles are as follows:

– Articles must explicitly state the study type (e.g., fault, quality, effort prediction) and provide evidence of metrics, methods, data sets exploited.
– Articles must exhibit a profound relation to OSS projects and take into account the aspects that can be attributed to the OSS community and projects.

Automated keyword search. At first a broad automated keyword search based on the title, keywords and abstract was performed to get the initial set of articles. Six digital libraries are searched within the time period of January, 1980 and 31st June, 2011. Search terms can be found in *http://literature-review.weebly.com/*.

Manual selection. To filter out the irrelevant ones from this set of articles, we performed a manual selection by reviewing the title, keywords, abstract and conclusion.

Reference checking. To ensure the inclusion of other relevant but missing articles, we performed a non-recursive search through the references of the 32 articles. Finally, we selected 36 articles (12 journal and 24 conference articles) for this review.

2.3 Attribute Framework

Attribute identification. The attribute set was derived based on: (a) The domain of the review and (b) The research questions. For this, a pilot study consisting of following activities was run: first, an exploratory study on the structure of 5 randomly selected articles was performed to identify initial set of attributes. Then, this list of attributes was refined further into a number of specific sub-attributes employing a through study of the same set of articles. The sub-attributes were then generalized further to increase their applicability. The final attribute set can be found at *http://literature-review.weebly.com/*.

2.4 Article Assessment and characterization

Appropriate set of attributes are assigned to the articles to effectively capture the essence in terms of the research questions and allow for a clear distinction between (and comparison of) the articles. The data colleciton table can be found at *http://literature-review.weebly.com/*.

3 Review result

3.1 Answering the Research Questions

Q1. What are the main focuses/purposes of the study?
Research interest toward predicting OSS projects predominantly dedicated to traditional defect/fault prediction studies (66%) , with minimal exploraion of the impact of OSS community in these prediction studies.
Q2. What are the datasets of OSS projects exploited in prediction? What else to be explored?
OSS development process produces repositories consisting of source code, bug reports, mailing lists, change logs, forums, wikis and so on. Due to such wide variety of data sources, we group them into different categories, such as, Contribution refers to bug reports, patches, feature requests.

According to our results, the highest utilized data sources are, source code version control systems (49%) and contribution (36%), with CVS repositories and the Bugzilla tool having the maximum exploration count. These sources are mainly used for fault or defect prediction. But the two sources, communication channels and knowledge sources (e.g., mail, chat, wikis), are yet to get attention confirming the fact that OSS community dynamics was not explored in prediciton studies.
Q3. What kinds of methods are used in OSS and prediction?
As can be seen from Figure 1(a), around 50% of the articles exploited statistical methods for prediction, whereas only 24% of the articles used machine learning algorithms. This result contradicts with the survey on fault prediction studies [3], where it was noted that machine learning algorithms are gaining more interest (increased from 18% to 66%) over statistical methods (decreases from 59% to 14%). This difference may be for two reasons, (a) the survey in [3] focused on fault prediction, whereas our survey covers the entire domain of prediction studies on OSS projects, and (b) OSS is relatively new area to explore for prediction studies. However, it will be promising to see the exploration of machine

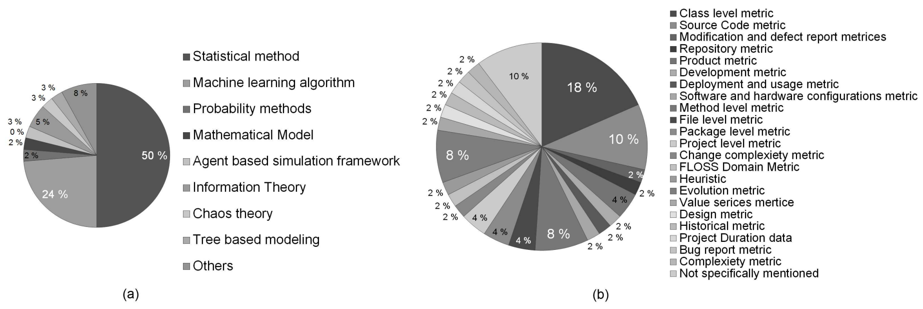

Fig. 1. (a) Methods employed for prediction (b) Metric suites studied

learning methods in OSS prediction studies which is also suggested in [3].
Q4. What kinds of metrics are used in predicting OSS projects?
The distribution of metrics which have been used in research for predicting OSS projects are shown in Figure 1(b). As shown in this figure, the class level metric and the source code metric suites got the highest priority for prediction (18% and 10% respectively). Among other metric suites, file level and package level, and project level metrics were also exploited. All these metric suits are prevalently used for fault prediction studies, hence confirms the findings presented in Q1.

3.2 Open Questions and Research Agenda

Q1. Are the generalizability of the prediction models hold across the domain of OSS projects? Or are they subjected to specific project(s)?
This research question evolved due to the fact that most of the reviewed articles (67%) admitted the necessity of external validity of the prediction models studied. To be specific, in [4] generalization of the findings was not done because the study is subjective and is dependent on how the errors are classified in the project. Again in [9], it is acknowledged that further replication across many OSS projects is required to establish the cross project validity of the prediction model. It is also noted that the prediction models are not general and are not applicable to different software systems [10]. Specially for defect prediction models there exists very little evidence on their cross project applicability [5]. Thus a comprehensive study on the generalizability issue of the prediction models across the domain of OSS projects is an area of future research.
Q2. Is the prediction accuracy of metrics remains consistent among the studies, or is there any contradictory results exist?
Each metric used for prediction, either being positively or negatively associated with prediction results. For example, in case of fault prediction, a metric signifies a module as either being faulty or not faulty. In either case, the metric's prediction recital is judged as a best, significant or bad predictor. In this regard, our review results show inconsistency on some metric's performance. For instance,

the metric LOC (Line of Code) was evaluated as a best or good predictor in [1][9], whereas in [11] it was noted as a bad predictor. Moreover, DIT (Depth of Inheritance Tree) was noted as a significant predictor in [6], but classified as a bad predictor in [1][4]. Possible causes behind these differences in results might be the variations in OSS systems [9], differences in implementations of the metrics [9], or different prediction models used. However, an indeepth investigation and resolution of such conflicting issues would be a future research agenda.

Q3. What metrics persist across the domains of the OSS projects? Researchers studied the effectiveness and accuracy of several metric suites using data from one or more OSS projects. Despite of their esteemed contribution in predicting OSS projects, they suffer from lack of generalizability due to diverse nature of OSS projects and the project specific nature of the metric suites. Also it is quite difficult to ensure the availability and quality of metric data, which makes the results incomparable [10]. Thus, a future research agenda would be to perform an indeepth analysis on (a) cross project validity of the studied metric suits, and (b) to propose methods to ensure the quality of metric data.

Q4. Contradiction or complementary?: (a) metric suit rely on a snapshot or (b) metric suit derived from the evolution of a project. Traditionally defect prediction models rely on metrics that represent the state of the software system at a specific moment in time [11].These metrics are used to capture a particular snapshot or release of a project to predict the next one. But metrics capturing changes over time in projects also play a significant role in prediction. For example, metrics presenting the software evolution were used to predict the need of refactoring [12] and quality of OSS projects with significant accuracy. Thus a future research direction would be to explore a comparative study for identifying either (a) which form of metrics are more suitable for prediction models in terms of accuracy, reproducibility, and generalizability, or (b) are these metrics complementary to each other and should be used in combination to get better prediction results.

Q5. What does the community structure predict for the OSS projects? What sets open source development apart from the traditional proprietary software is the developer community behind it. Although the social structure and communication of OSS communities have gained significant research interest, the research efforts to the community in relation to prediction appear quite the opposite. Evolution of communities is of interest starting from the paper introducing the community structure [13] but our search did not find much focus on community evolution tied to prediction. In [14] the authors investigate the impact of social structures between developers and end-users on software quality and their results give support to thinking that social structures in the community do hold prediction power in addition to the source code centric approaches. It is also suggested that combining metrics focusing on code and social aspects work as a better prediction model than either alone. This gives support that the question has research value and is worth looking into further: what does the community and the community structure predict for the software?

4 Discussion

SLR concerning software fault prediction was first conducted by [3] and was extended with new results in [7]. However these works were limited to fault prediction of closed source projects and fall short of exploring OSS domain.

This SLR will help researchers to investigate prediction studies from the perspective of metrics, methods, datasets, tool sets in an effective manner. Future research should focus on establishing external validity and consistent accuracy of prediction models, incorporation of social aspects of OSS projects, and building tool support to automate the prediction process.

References

1. Gyimothy T, Ferenc R, Siket I (2005) Empirical Validation of Object-Oriented Metrics on Open Source Software for Fault Prediction. TSE 30(10):897–910
2. Kitchenham BA, Pretorius R, Budgen D, Brereton OP, Turner M, Niazi M, Linkman S (2010) Systematic literature reviews in software engineering- A tertiary study. Information and Software Technology 52(8):792-805
3. Catal C, Diri B (2009) A systematic review of software fault prediction studies. Expert Systems with Applications 36(4):7346–7354
4. Shatnawi R, Li W (2008) The effectiveness of software metrics in identifying error-prone classes in post-release software evolution. TSE 81(11):1868–1882
5. B. Turhan and T. Menzies and A. B. Bener and J. D. Stefano (2009) On the relative value of cross-company and within-company data for defect prediction. ESEM 14(5):540–578
6. Subramanyan R, Krishnan MS (2003) Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects. TSE 29(4):297–310
7. Catal C (2011) Software fault prediction: A literature review and current trends. Expert Systems with Applications 38(4):4626–4636
8. Capiluppi A, Adams PJ (2009) Reassessing Brooks Law for the Free Software Community. In: IFIP AICT, pp.274-283
9. English M, Exton C, Rigon I, Cleary B (2009) Fault detection and prediction in an open-source project. In: PROMISE
10. Askari M, Holt R (2006) Information theoretic evaluation of change prediction models for large-scale software. In: MSR, pp.126-132
11. Knab P, Pinzger M, Bernstein A (2006) Predicting Defect Densities in Source Code Files with Decision Tree Learners. In: MSR, pp.119-125
12. Ratzinger J, Sigmund T, Vorburger P, Gall H (2007) Mining Software Evolution to Predict Refactoring. In: ESEM, pp.354-363
13. Nakakoji K, Yamamoto Y, Nishinaka Y, Kishida K, Ye Y(2002) Evolution Patterns of Open-Source Software Systems and Communities. In: IWPSE, pp.76-85
14. Bettenburg N, Hassan AE (2010) Studying the Impact of Social Structures on Software Quality. In: ICPC, pp.124-133