Effort-Oriented Methods and Tools for Software Development and Maintenance for Mobile Apps

Gemma Catolino
University of Salerno, Department of Computer Science, Italy

ABSTRACT

The present research project aims to propose methods and tools for mobile applications development and maintenance that rely on effort information (estimations). Specifically, we will focus on two main challenges to overcome existing work: (i) conceiving effort estimation approaches that can be applied earlier in the development cycle and evolve through the development process (ii) prioritizing development and maintenance tasks by relying on effort estimation information.

KEYWORDS

Effort Estimation, Maintenance and Evolution, Mobile Apps

ACM Reference format:
Gemma Catolino. 2018. Effort-Oriented Methods and Tools for Software Development and Maintenance for Mobile Apps . In *Proceedings of 40th International Conference on Software Engineering , Gothenburg, Sweden, May 27-June 3, 2018 (ICSE '18 Companion),* 2 pages.
DOI: 10.1145/3183440.3183457

1 CONTEXT

The wide diffusion of smartphones and tablets has led to an ever increasing demand of mobile applications. Although these apps are generally smaller than traditional systems (e.g., in terms of size or duration of life cycle), their complexity is steadily increasing. So, it is necessary to provide a proper monitoring of the quality of mobile apps during both development and maintenance phases. The availability of suitable effort estimation approaches would be very useful or anticipating costs and staff resources, project planning and monitoring [2, 11]. Unlike traditional systems, there are not many studies presenting approaches for effort estimation within the mobile context. Moreover, the majority of them concerns with estimating mobile app size (e.g., [1, 8]), that represents only one of the possible cost drivers. In addition, these approaches can be used only when the requirement analysis has been completed.

To deal with this issue, it is needed to identify factors that provide an *early* effort estimation, moving the effort estimation task towards a prior phase of software life cycle [10].

Another important limitation of the current state of the art is represented by the lack of approaches exploiting effort-related information (e.g., number of bugs contained in a source code class) to support developers in their daily activities. Indeed, most of the techniques mentioned above can be applied for the estimation of the effort needed to develop an *entire project*, while fine-grained tools able to drive developers during the development and maintenance tasks are missing.

Following these considerations, the research project and interest cover different areas and topics of Software Engineering and Project Management in the context of mobile apps, which may be the means to reach the specific goals described later in this paper.

2 RESEARCH STATEMENT

2.1 Estimating Effort Estimation through software life-cycle

A common problem of project managers is estimating as soon as possible size and complexity of an application to be developed, so that they can correctly estimate the cost needed for developing it and schedule the activities. While some effort prediction models have been proposed in the past [1, 8], their common limitation is represented by the fact that they can be only exploited when requirements have been *already specified.* With the aim of better support project managers in the effort estimation activity, this challenge firstly aims at identifying factors/metrics that can be gathered before the beginning of a project and that can be effectively exploited to estimate effort. Then, we will try to understand how it is interesting understand how these cost drivers evolve during the software development and maintenance process.

Main Goal. Identification and evaluation of cost drivers relevant in different phases of mobile apps software development. Building and presenting a dataset with effort data of mobile apps.

Estimating Effort Earlier. Mendes et al. [10] systematically investigated the factors extractable in the early phase of software development that influence the effort for the development of web applications. In our context, we tried to identify a set of metrics that allow the estimation of the effort during a preliminary stage of the development of mobile software, i.e., when the functional requirements are not have still been defined [6].

Preliminary Results. Following the methodology proposed by Mendes et al., starting from the online quote forms available on the web by app companies, we systematically collected and analyzed them. In the first place, we manually checked all the discovered quotes with the aim of discarding the false positive ones, i.e. the ones that reported only generic information without specifying exact data to use for estimation purposes. Then, we extracted a total of 48 factors generally used by companies to estimate the effort of a project in the early phase of mobile app development. In the second place, we surveyed four project managers having a good experience in managing and developing mobile apps with the aim of verifying the validity of the factors identified. As a result, starting from 48 factors, 36 of them have been confirmed by the experts as the ones having a higher impact on the effort of a project, while 12 have been

ICSE '18 Companion, Gothenburg, Sweden
© 2018 Copyright held by the owner/author(s). 978-1-4503-5663-3/18/05...$15.00
DOI: 10.1145/3183440.3183457

discarded. The experts also suggested the introduction of new 5 factors that were not considered in the previous step. Due to space limitation, the set the factors is available on the online appendix [3] as well more details of paper are available on [6].

Estimating Effort during Software Maintenance. As widely known, software life cycle does not end with the delivery of the first working version of the system, but rather it proceeds until its decommissioning. Thus, any project has an important and expensive and cost maintenance phase. In this context, we would like to propose a novel effort estimation model able to predict the class of cost (based on the distribution of Lines of Code (LOC)) of a generic maintenance task of an app using a Likert scale (e.g., from very low to very high). Previous work, in the context of traditional applications, investigated potential maintenance cost drivers and built prediction models that support project managers in estimating the likely effort needed to issue a new release of the system [12] (i.e., in terms of man/hours, LOCs to modify, completion time). However, we believe that current state of the art suffers some important limitations due the (i) granularity of the prediction (e.g., release level), (ii) lack of generality, since only approaches working for specific maintenance activities have been proposed (e.g., bug resolution), while none of them can estimate the effort of a generic maintenance task. Moreover, the estimation of effort depends on programming language, and existing effort estimation models do not consider developer-related factors (e.g., experience) whose potential for effort estimation has been empirically shown [5, 7]. So, our conjecture is to identify factors relevant within the maintenance task effort estimation, based on the existing literature, and understand how the use of developer related factors affects the prediction of the effort of a task.

2.2 Prioritizing Software Engineering Tasks using Effort-related Information

Continuous releases, the pressure to deliver new functionalities as soon as possible (especially in the mobile context), and the need to control that new changes do not introduce bugs are just some of the challenges that developers face on a daily basis. During such activities, several decisions have to be made: for instance, which classes should be tested first, which classes require more attention with respect to code quality improvement, etc. In this context, the research community is devising several tools to ease maintenance and evolution of mobile apps, but there is still a lack of tools helping developers in timely spotting possible bugs. Toward this direction, a promising area is represented by software bug prediction: indeed, time spent to x bugs can be considered as an important effort-related information that can be exploited to provide developers with more powerful recommenders.

Main Goal. Definition of effort-aware tools to support developers during development and maintenance phases.

Bug Prediction Model. In this area of the research we preliminarily evaluated the usefulness of the metrics proposed by Kamei et al. [9] for Just-In-Time (JIT) bug prediction in the context of mobile app development [4]. This is a more practical alternative to traditional bug prediction techniques, because it provides feedback at commit-time. In particular, we performed the study by taking into account five mobile apps with different size and scope; then, we applied a feature selection algorithm in order to avoid multicollinearity. Once we had defined the set of independent variables, we built a Logistic Regression model assessing the performance using a 10-fold cross validation strategy. The results showed how this model is not immediately exploitable in the context of mobile apps. Indeed, the value of the recall was around 25%. This implied that the model was able to discover only a limited number of real bugs occurring in an apps, so more analysis are necessary. In our work, we plan to further investigate cost drivers and metrics to improve the prediction capabilities of JIT models.

3 OPEN ISSUES AND FUTURE DIRECTIONS

Open Issue #1 - Toward an early effort model prediction. Based on the results of this challenge, we are already conducting a further evaluation on the usefulness of the factors focusing on the opinion of the developers. Indeed, while the Project Managers plan the effort of a project, the developers actually spend the effort. To this aim, we are starting the preparation of mobile project data entry forms to gather data on mobile projects worldwide, that will be used to train and test the effort prediction model.

Open Issue #2 - Toward an effort model prediction in different phases of development. Starting from the results of the first open issue, we plan to conduct industrial surveys in order to understand which factors are important for estimating the effort in a certain phase of software development. The output will consist of a set of factors for different development phases.

Open Issue #3 - Cross-project Just-in-Time Bug Prediction. Based on the results of our previous study [4], our idea is to explore the applicability of a cross-project strategy to the mobile context. Moreover we conjecture that the usage of different classifiers (e.g., logistic regression or classification trees) with the usage of a large number of apps could influence the performance of the model.

REFERENCES

[1] Allan J Albrecht. 1979. Measuring application development productivity. In *Joint SHARE/GUIDE/IBM application development symposium.*

[2] Lionel C Briand and Isabella Wieczorek. 2002. Resource estimation in software engineering. *Encyclopedia of software engineering* (2002).

[3] Gemma Catolino. *Effort-Oriented Methods and Tools for Software Development and Maintenance for Mobile Apps.* Technical Report.

[4] Gemma Catolino. 2017. Just-in-time bug prediction in mobile applications: the domain matters!. In *Proceedings of the 4th International Conference on Mobile Software Engineering and Systems.*

[5] Gemma Catolino, Fabio Palomba, Andrea De Lucia, Filomena Ferrucci, and Andy Zaidman. 2017. Developer-related factors in change prediction: an empirical assessment. In *Proc. of the 25th International Conf. on Program Comprehension.*

[6] Gemma Catolino, Pasquale Salza, Carmine Gravino, and Filomena Ferrucci. 2017. A set of metrics for the effort estimation of mobile apps. In *Proceedings of the 4th International Conference on Mobile Software Engineering and Systems.*

[7] Dario Di Nucci, Fabio Palomba, Giuseppe De Rosa, Gabriele Bavota, Rocco Oliveto, and Andrea De Lucia. 2017. A Developer Centered Bug Prediction Model. *IEEE Trans. on Softw. Engineering* (2017).

[8] Reiner Dumke and Alain Abran. 2011. *COSMIC Function Points: Theory and Advanced Practices.* Auerbach Publications.

[9] Yasutaka Kamei and others. 2013. A large-scale empirical study of just-in-time quality assurance. *IEEE Transactions on Software Engineering* (2013).

[10] Emilia Mendes, Nile Mosley, and Steve Counsell. 2003. Investigating early web size measures for web cost estimation. In *Proceedings of EASE'2003 Conference.*

[11] Ian Sommerville. 2006. *Software Engineering: 8th Edition.* Addison-Wesley Longman Publishing Co.

[12] Hong Wu, Lin Shi, Celia Chen, Qing Wang, and Barry Boehm. 2016. Maintenance effort estimation for open source software: A systematic literature review. In *Software Maintenance and Evolution (ICSME).*