The Need for Effort Estimation Models for Open Source Software Projects

Jai Asundi
Information Systems and Operations Management
School of Management, University of Texas at Dallas
asundi@utdallas.edu

ABSTRACT
Open source software(OSS), be it products or tools, are being adopted at a fairly rapid pace in commercial organizations. In fact many firms such as IBM and Sun are even 'opening' up their once proprietary software products and making the source code available. This phenomenon may have a profound effect on the various software engineering methodologies and practices as well as project management activities. Given the difficulty in managing resources in closed source projects, planning and delivery for OSS projects will be an even bigger challenge. In this position paper, we describe the need for new effort estimation models for the development of OSS projects and how this will be required for future project management activities. We outline some of the guidelines to build these cost estimation models and some issues that arise in the verification and validation of these cost models.

Keywords
Open Source Software, Maintenance, Effort Estimation, project management

1. INTRODUCTION
Open source software(OSS) products or tools are being adopted at a fairly rapid pace in commercial organizations. Software products like the Linux operating system, the Apache Web-server, and MySQL database server are considered to be on par or even better than other functionally comparable commercial software product offerings. Industry experts, Open Source developers and Researchers have been examining the phenomenon of OSS and the new methods of software development[4]. A significant area of research has been in examining the motivations of the open source developers to contribute freely[3]. OSS projects also lend themselves to mining of source-code repositories and mailing lists to examine issues such as design, leadership and conflict resolution[9,10, 11].

Software firms (like IBM, Sun and Apple, BEA) are making significant investments and contributions to OSS products. This is being executed in a number of ways. A few of them are:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Open Source Application Spaces: Fifth Workshop on Open Source Software Engineering (5-WOSSE) May 17, 2005, St Louis, MO, USA.

Copyright 2005 ACM 1-59593-127-9 … $5.00.

1. By opening up their closed source software products (IBM- Cloudscape database, Jikes compiler, Sun- Star Office, SunOS, Apple – Darwin 7.0 etc.)[6][7][8]
2. By providing development resources (IBM developers on Apache Webserver)
3. Or by providing monetary resources to support OSS project related activities.[5]

While these contributions are interesting and welcome, acceptance of the OSS form of development into traditional software development organizations may have a profound effect on the various existing software engineering methodologies and practices as well as project management activities. Given the difficulty in managing resources in closed source projects, planning and delivering projects that are based on an open source community can be a bigger challenge. Resource allocation and budgeting will be harder and without a rigorous basis.

For this reason, in this position paper, we argue that we need to develop effort estimation models for the development of OSS products. For the purpose of this paper, we assume that cost and effort are synonymous and use these terms interchangeably. The paper is organized as follows. Section 2 outlines the some issues with effort estimation and existing software effort estimation models and techniques. Section 3 describes some of the issues with OSS that need to be considered when building cost models and Section 4 concludes by providing some generic guidelines and plans for future work.

2. EFFORT ESTIMATION MODELS
In this section we outline some of the general cost/effort estimation models, some software effort estimation models and the issues in using these models for OSS projects.

Effort(or cost) estimation in software engineering projects can be categorized into three levels of detail and accuracy: Order of Magnitude, Semi-detailed/ Conceptual estimates and Detailed. The effort required in getting an estimate itself increases with the accuracy of estimate required. In most engineering projects, most cost/effort estimates are conservative and usually lower than actual[12]. In the design and building of large complex systems, effort/cost estimation models can be a difficult task due to the following reasons:

1. System of that size/type has never been built before
2. New/Unproven technologies are being put to use
3. Productivity of personnel has high variance

Measurement difficulties and the high variability with the associated measures are peculiar to software based systems. Measurement of outputs and effort are very subjective and there is considerable debate regarding the same. According to Jalote [13] a software product is an entirely conceptual entity and thus there are no physical or electrical laws that govern software engineering. Boehm[1] adds that software development requires creativity and cooperation of human beings whose individual and group behavior is generally hard to predict.

In most engineering systems, historical data is used as a basis for cost/effort estimation for future projects. Unfortunately, in most cases, especially for software products, reliable data are difficult if not impossible to find. Most of the time the development and construction technologies underlying the software product undergo significant changes from one generation to the next. This means that in engineering software, we are frequently building and maintaining systems of unprecedented size and complexity and with rapidly changing tools. This wide variability in system type and nature results in inaccurate estimates.

Existing software effort estimation models could be categorized into the following (not exhaustive) types:

1. Analogy costing: The resource estimates are developed based upon past experience with similar systems. A pair wise comparison with a similar project on a component-wise basis is used to obtain an overall estimate
2. Delphi costing: Resource estimates are developed using a team of experts. The team of experts estimate resources under similar assumptions and then agree upon a consensus estimate
3. Parametric model costing: Resource estimates are developed using prediction models which mathematically relate effort and duration to the parameters that influence them. These models are built up using regression analysis on available data

Parametric models are amongst the most popular. Simplistically, effort is estimated as:

$$Effort = a\ (Size)^p$$

Where a represents the impact parameters, Size is the measure of output (SLOC or function points) and p is exponent relating size to effort. The parameters a and p are estimated from historical data. Well known models of this type are COCOMO, SLIM and PRICE-S.

One of the popular and most elaborate effort estimation models for new software development is the COCOMO[Boehm81]. The COCOMO model (which consists of three submodels: Basic, Intermediate and Advanced) can be written as:

$$E = K\ S^{\alpha} \prod C_i$$

Where K and α are parameters on the mode the software system is developed and the fifteen C_i's are the cost drivers [14].

If necessary, OSS projects one can only use Delphi costing – i.e. the use of experts to determine the resource requirements in a project. Analogies are hard to find and there are no currently calibrated parametric models.

There are many drawbacks to using existing effort estimation models for OSS projects. For one, the existing models are based on data collected from commercial "closed" source projects. The platforms and tools that are used to develop the software differ from those used in OSS projects. Most of the data collected for existing software products have been for long term new development projects rather than maintenance projects. In OSS projects a code-base already exists, developers will be fixing existing bugs and adding functionality in a piecemeal fashion. We can thus argue that most OSS projects are in a maintenance-enhancement phase of development. Models such as the COCOMO assume proportionality of effort when it comes to the maintenance phase of a software project. This assumption we feel is inadequate for the treatment of OSS projects, due to the nature and complexity of the maintenance task in an open source context.

3. OSS EFFORT ISSUES

In this section we will outline some of the issues relating to OSS projects that need to be kept in mind while building an effort estimation model.

Like any software development project, effort is not evenly distributed across all participants. An analysis of the modification requests in the Apache server project (by participants) shows that more than 91% of the modification requests are being made by 10% of all contributors.

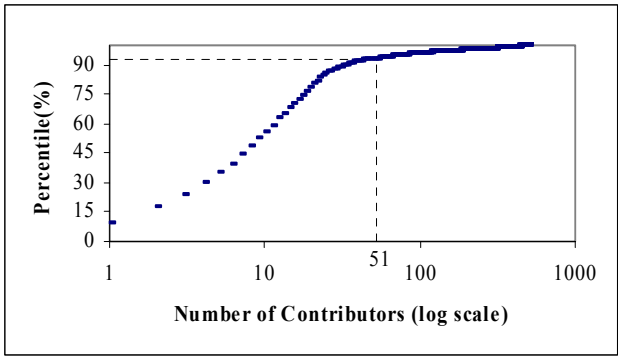

Fig 1: Cumulative distribution of modification requests

This skewed nature may seem a little extreme for any closed source project and thus will have a profound effect on resource allocation in an OSS project.

An issue that we will face in developing effort estimation models for OSS projects is the validation of the model itself. Unlike commercial organizations, few OSS developers keep track of the effort they spent in fixing bugs or developing new functionality. All we may have are time-stamps for when the bug was identified to the time it was considered 'closed'. Correlating this time interval with the effort may lead to wildly fluctuating estimates and thus an unreliable model.

Another factor that may influence our effort estimation model is the possibility of collaboration amongst individuals that are far flung across the world. While this imposes restrictions in communication it could lead to shorter cycle-times for fixing certain bugs (almost akin to the 24X7) maintenance model adopted by software service organizations in India[15]. This is almost dependent on the culture within the project we are considering rather than any OSS project.

In OSS projects, majority of the effort seems to be expended in routine maintenance of the existing system or rather fixing bugs. Models such as the COCOMO use SLOC as a measure of size (or rather output). In a maintenance context, effort estimates are a function of the code that was changed. The question that arises is: Is SLOC a good measure of output in a maintenance context? While the size of the code-base is correlated to the complexity of the code, the modification of a small number of lines does not by any means indicate that less effort was spent thinking about and formulating a solution to a problem that a modification that required modification of a relatively larger number of lines of code.

4. DISCUSSION AND FUTURE WORK

We believe that effort estimation models for OSS projects will be important in the future as more commercial organizations adopt the processes and open their development to the community to modify and improve. Organization will need means by which they can estimate the amount of resources they must invest in the project in order to obtain the necessary deliverables (be it in improved functionality, lower defects or improved security)

The existing effort estimation models are inadequate for the purpose and thus we need to develop new models for OSS projects. In the previous section we outlined some of the issues that need to be taken into consideration whilst developing an OSS effort estimation model.

We plan to work on developing an OSS effort estimation model that will take into consideration some of the issues raised here. While parametric models are not completely ruled out, we see that in the case of OSS projects, we may have to develop "Activity Based Costing" type estimation models. In this kind of model one breaks down the task at hand into unit activities for which one can easily estimate the effort required.

In future work, we also plan to use the available time-stamp data for bug fixes and run controlled experiments in fixing bugs to correlate these time intervals with possible effort estimates. This may help us in validating some parts of our model.

We see this as a first step towards a new form of software development and look forward to interesting ideas from other researchers and practitioners.

5. ACKNOWLEDGMENTS

Thanks to faculty at the University of Texas at Dallas for useful discussions on this topic.

6. REFERENCES

[1] Boehm, B. *Software Engineering Economics*, Prentice Hall, New Jersey, 1981.

[2] Feller, J. Fitzgerald, B. 2002. Understanding Open Source Software Development Addison Wesley, London, England.

[3] Lerner, J., Tirole, J. (2002). "Some Simple Economics of Open Source," *Journal of Industrial Economics*, 52, 197-234

[4] Raymond, E. 1999. *The Cathedral and the Bazaar*, O'Reilly, Sebastopol, CA.

[5] "IBM puts cash behind Linux push", http://news.bbc.co.uk/1/hi/technology/4276287.stm downloaded on 2/19/2005.

[6] "Star Office released in Largest Open Source Project", http://linuxtoday.com/news_story.php3?ltsn=2000-10-13-002-21-NW-DT-SW D downloaded on 2/19/2005

[7] "Apple releases source code for Darwin 7.0" http://www.newsforge.com/os/03/10/28/2247203.shtml?tid=6&tid=82&tid=94 downloaded on 2/19/2005

[8] "IBM to release Java Database to Open-Source Group" http://www.eweek.com/article2/0,1759,1630856,00.asp, downloaded on 2/19/2005

[9] Jensen, C and Scacchi, W. "Collaboration, Leadership, Control, and Conflict Negotiation in the NetBeans.org Software Development Community," *Proc.of the 38th. Hawaii International Conference on Systems Sciences,* Kona, HI, January 2005

[10] Jensen, C and Scacchi, W. "Data Mining for Software Process Discovery in Open Source Software Development Communities," *Proc. Workshop on Mining Software Repositories*, 96-100, Edinburgh, Scotland, May 2004

[11] Gasser, L. Scacchi, W., G. Ripoche, and B. Penne "Understanding Continuous Design in F/OSS Projects," *16th. Intern. Conf. Software & Systems Engineering and their Applications*, Paris, December 2003.

[12] W. R. Park and D. E. Jackson, *Cost Engineering Analysis: A Guide to Economic Evaluation of Engineering Projects*, John Wiley, New York, 1984.

[13] Pankaj Jalote, An Integrated Approach to Software Engineering, Springer-Verlag, New York, 1991.

[14] Hu, Qing , R. Plant and David Hertz, "Software Cost Estimation Using Economic Production Models," *Journal of Management Information Systems*, Vol.15, No.1, pp. 143-163, Summer 1998

[15] Arora, A., Arunachalam, V.S., Asundi, J.M., and Fernandes, R., "The Indian Software Services Industry," *Research Policy* (30) 8, 2001, pp.1267-1287